

bd Systems®  
TCD20000103A  
30 April 2000

Contract No.  
NAS8-40604  
Final Technical Report

**Final Report**  
  
**for**  
  
**Mathematical Model Development and Simulation Support**  
**Contract No. NAS8-40604**

**Submitted To:**  
**Procurement Office**  
**Attn: Amy Campbell**  
**George C. Marshall Space Flight Center**  
**Marshall Space Flight Center, Alabama 35812**

**Prepared By:**  
  
**Ronald C. Francis**  
**Manager, GN&C Group**  
**bd Systems, Inc.**

**and**  
  
**Dr. Patrick A. Tobbe**  
**Dynamic Concepts, Inc.**

**April 30, 2000**

## TABLE OF CONTENTS

|  |           |
|--|-----------|
| <b>TABLE OF CONTENTS</b>   | <b>I</b>  |
| <b>1.0 INTRODUCTION</b>  | <b>1</b>  |
| <b>2.0 TASK SUMMARY</b>  | <b>1</b>  |
| <b>2.1 Task A: Contact Dynamics / 6-DOF Facility Support</b>                     | <b>1</b>  |
| ROCKET RMS Flight-to-Sim Validation  | 1         |
| 6DOF Hydraulic Table Characterization  | 1         |
| 6DOF Facility Finite Element Modeling  | 2         |
| ROCKET RMS Sim-to-Sim Validation   | 2         |
| Porting ACDS and TCDS from Alliant to SGI  | 2         |
| Contact Force Model Characterization Tests                                       | 2         |
| SRMS Math Model Support  | 2         |
| Porting ROCKET from Alliant to SGI Challenge XL                                  | 2         |
| Porting DEBERTH and TWO BODY from Alliant to SGI                                 | 3         |
| Hardware/Software Integration of RMS and 2BODY                                   | 3         |
| Implementation of the Digital SPA and POHS into ROCKET                           | 3         |
| ROCKET Modifications for ECP275  | 4         |
| ROCKET SRMS Sim-to-Sim Validation with the digital SPA and POHS                  | 4         |
| Compensation Curve Update  | 5         |
| Implementation of Artificial Damping and Force Filtering into ROCKET             | 5         |
| CBM Multi-Stage Capture Testing  | 5         |
| Support 6DOF Testing with Astronauts and Trainers                                | 5         |
| ROCKET Post Processing Update  | 6         |
| SSRMS Math Model   | 6         |
| <b>2.2 Task B-1: Flight Robotics Laboratory / DOTS Support</b>                   | <b>7</b>  |
| DOTS Software Upgrades   | 7         |
| <b>3.0 SUMMARY</b>   | <b>9</b>  |
| <b>REFERENCES</b>  | <b>10</b> |
| <b>APPENDICES</b>  | <b>11</b> |
| 1. ROCKET SRMS Flight-to-Sim Validation Test Report                              |           |
| 2. ROCKET SRMS Sim-to-Sim Validation Test Report                                 |           |
| 3. TH60020A, RMS Math Model Upgrades and Simulation Validation Support           |           |
| 4. TH70039A, Attachment, DCI TM#081097, DOTS RMS Script Commands                 |           |
| 5. TH70044A, Attachment, DCI TM#092997, ROCKET Flight-to-Sim Validation Runs     |           |
| 6. TH70052A, Attachment 1, DCI TM#101497-1, ROCKET Flight-to-Sim Validation Runs |           |

7. TH70052A, Attachment 2, Bridge Encoder Test Report
8. TH80008A, Attachment 1, DCI TM#011398-1, 2BODY Code Modifications and Test Runs
9. TH80008A, Attachment 2, DCI TM#011398-2, Deberth Code Modifications and Test Runs
10. TH80008A, Attachment 3, DCI TM#012798-1, 2BODY Real-Time Simulation Integration and Validation
11. TH80023A, Attachment 1, DCI TM#052398-1, 6DOF Facility Hardware / Software Integration
12. TH80034A, Attachment, DCI TM#082698-1, Digital Servo Power Amplifier Model
13. TH80042A, Attachment 1, DCI TM#103198-1, RMS Simulation POHS Upgrade
14. TH90030A, Attachment 1, DCI TM#020499-1, RMS Simulation CBM Verification Testing
15. TH90120A, Attachment, DCI TM#052599-1, RMS Simulation Support
16. TCD20000049A, Attachment, DCI TM#122799-1, RMS Simulation Support

## 1.0 INTRODUCTION

This report summarizes the work performed under contract NAS8-40604. This work was initially awarded to bd Systems, Inc., Control Dynamics Division on November 27, 1995. This contract primarily supports the Six Degree of Freedom Motion Facility (6DOF) and Flight Robotics Laboratory (FRL) at the George C. Marshall Space Flight Center (MSFC). The contract was modified and extended with several increments of funding to continue support for these and other activities. All work under this contract has been completed and documented [References 1-16]. The Appendices contain the reports listed in the references.

Section 2.0 of this report summarizes the various activities performed by bd Systems and its subcontractor, Dynamic Concepts, Inc. Summary remarks are presented in Section 3.0.

## 2.0 TASK SUMMARY

### 2.1 Task A: *Contact Dynamics / 6-DOF Facility Support*

#### ROCKET RMS Flight-to-Sim Validation

bd Systems supported MSFC in improving the ROCKET SRMS model validation score when compared to flight data. The ROCKET simulation was run with various friction and gearbox damping models. A problem with the deadband on the joystick inputs being set to zero was corrected for the man-in-the-loop validation cases. bd Systems ran the validation cases with the changes to the man-in-the-loop logic, new real time bending modes, and strain gauge damping matrices. The model used for these runs was transferred to Alli and integrated with concrete. The new real time bending modes for the arm linkages were computed using a modified form of the Craig-Bampton technique. Simulation runs with the Boeing model were used to reduce the number of these substructure modes required for real time operations. After completing the validation runs, bd Systems assisted Mark Slone of NASA MSFC in grading the validation runs and preparing the formal validation report for the RMS Math Model Working Group. (Reference 1, ROCKET SRMS Flight-to-Sim Validation Test Report, NASA MSFC, April 1996) In support of the formal SRMS math model validation process, bd Systems upgraded and enhanced the SRMS simulation. Reference 3 is the documentation of these upgrades.

#### 6DOF Hydraulic Table Characterization

bd Systems participated in a series of system identification tests to characterize the hydraulic table. These tests consisted of driving the legs individually with random noise through the analog channels and exciting the table with noise commands through the Alliant in table space. The data taken during these tests was used in the construction of a simulation model of the hydraulic table and associated control system.



### 6DOF Facility Finite Element Modeling

bd Systems worked in correlating the finite element model of the facility support structure with the modal test results. The finite element model of the facility support structure was upgraded to match test data. This model was then integrated into the facility analytical contact dynamics simulation TCDS.

### ROCKET RMS Sim-to-Sim Validation

bd Systems supported Mark Slone of NASA MSFC in running the ROCKET SRMS sim-to-sim validation runs, grading the validation runs and preparing the formal validation report for the RMS Math Model Working Group. (Reference 2, ROCKET SRMS Sim-to-Sim Validation Test Report, NASA MSFC, February 1997) The ROCKET SRMS results were graded against SPAR's All Singing/All Dancing (ASAD) simulation results. For this effort, the RMS code was modified to incorporate bending effects into the POR data.

### Porting ACDS and TCDS from Alliant to SGI

bd Systems supported the transfer to and conversion of the serial versions of the RMS and berthing mechanism simulations, ACDS and TCDS, on the SGI Challenge XL computer, *uqbar*. The simulation output routines were modified to be compatible with the MSFC plotting package. Makefiles were also constructed for both simulations on *uqbar*.

### Contact Force Model Characterization Tests

bd Systems supported Boeing and MSFC in a series of contact force model characterization tests in the 6DOF. This data was used to correlate the contact force model with future HWIL runs.

### SRMS Math Model Support

bd Systems supported MSFC in its participation in the SRMS Math Model Working Group through both travel to JSC and teleconferences with JSC on the SRMS model.

### Porting ROCKET from Alliant to SGI Challenge XL

bd Systems participated in the SGI Real Time Programming and Parallel Programming classes at MSFC in preparation of conversion of ROCKET to the SGI computer, *uqbar*. Work was done to decrease the cycle time of the RMS simulation. A transformation matrix was corrected in the Boeing payloads which results in smoother analytical backdrive forces and moments. This modification was incorporated into the real time simulation. A problem was discovered in the

post processing of the backdrive force subroutine used in the 6DOF software. A problem was discovered by bd Systems in the initialization of the parallel code. The non-real time code was updated and tested in preparation of moving the real time code from the Alliant computer to the SGI Challenge computer.

The common blocks for the shared memory used between processes were converted into structures to eliminate the long DT. The resulting changes to the variable names were completed. The new makefiles were created and the individual source code files were compiled and compile time errors corrected. The NASA written C executive source code and the bd Systems modified Fortran code were linked together and an executable generated. The new ROCKET executable was tested by performing validation runs. All of the validation runs were successfully completed. Reference 5 contains the documentation of the ROCKET flight to simulation validation runs that were performed as a validation of the new version of ROCKET on *uqbar*. When performing the ROCKET flight to simulation validation runs, a high frequency oscillation was discovered in the strain gage loads for some runs. Reference 6 documents the corrections to the ROCKET software to eliminate this effect.

#### Porting DEBERTH and TWO BODY from Alliant to SGI

bd Systems supported the porting and conversion of the DEBERTH and 2BODY software to the SGI computer *uqbar*. New Fortran structures were developed and integrated into the hardware interface routines. Documentation of the 2BODY code modifications and test runs is provided in Reference 8. Documentation of the DEBERTH code modifications and test runs is provided in Reference 9. The 2BODY software was integrated with the new parallel enabled executive and validation runs were conducted with the integrated simulation. Documentation for the 2BODY real-time simulation integration and validation is provided in Reference 10.

#### Hardware/Software Integration of RMS and 2BODY

bd Systems supported the hardware/software integration of the SRMS and 2BODY on the SGI computer, *uqbar*, with the 6DOF facility. The following tests were conducted by bd Systems during this integration task: Functional panel tests, rope pull tests with both the 2-Body and RMS simulations, compensation tests with the coil springs (translation) and leaf springs (rotation), RMS runs with the coil springs, 3 latch runs with brakes off for 3 different payloads. Documentation of this integration effort is in Reference 11. Software development of the joystick and graphics panel for communication with the existing RMS flight software and model was also completed and tested.

#### Implementation of the Digital SPA and POHS into ROCKET

bd Systems and Dynamic Concepts reviewed the SPA documents, SPAR-RMS-SG.1936 Issue G, SPAR-RMS-TM.2163 Issue B and SPA Software Design Document #823492 Revision P and

implemented the digital SPA model into the ROCKET simulation. They also developed a model of the MCIU (manipulator controller interface unit) to implement transport time delays between the general purpose computer and the RMS joint servos and integrated this model into ROCKET. Reference 12 documents the changes to the ROCKET simulation for the addition of the digital SPA. Selected flight to simulation validation cases were run with analog and digital SPA models and MCIU time delays for validation of the SPA additions.

The RMS model in the ROCKET simulation was updated to include the Position Orientation Hold Select (POHS) capability in the flight software. Reference 13 documents the changes for the POHS upgrade. As part of this activity, all modes of operation of ROCKET were re-organized to operate under flight software control. Several flight software and simulation initialization routines were modified or deleted to accomplish this goal. These activities were necessary to meet the requirements of integrated CBM test 275.

### ROCKET Modifications for ECP275

A review of the RMS flight software was performed to identify modifications required for ECP 275. The Single Joint and Idle modes of operation, including mode transition logic, were added to the 6 DOF HWIL real time software and interfaced to the RMS graphics panel. Functional tests were performed to demonstrate this new capability. Testing and modifications of the new man-in-the-loop (MIL) flight software was also completed and demonstrated to Boeing. The V20 interfaces with 2BODY and ROCKET were refined. The software only verification tests of 2BODY in the V20 facility were conducted. The RMS control system I-LOADS for Flight 3A and 5A configurations were reviewed.

In preparation for ECP 275, three major modifications were made to ROCKET for MIL operation: addition of analytical contact force models, integration of an event scheduler / paper pilot, and calculation of RSAD and spec panel 169 display variables. Reference 15 documents these modifications. The guide-to-guide, plunger and RTL contact models from ACDS were added to the real time ROCKET simulation under the *DYN* process. Using these contact models is optional, and is controlled based on user input. Analytical contact force models in ROCKET were updated to correlate with the HWIL test results. The fly-to and fly-from modes of MIL were modified to define new reference frames for the RMS joystick commands at initialization only. ROCKET was upgraded to incorporate a paper pilot event scheduler. A more detailed MCIU communications model was integrated into ROCKET. Comparison runs were made to correlate ACDS and ROCKET. Over 40 HWIL paper pilot test runs were made in the 6DOF with LAB and PMA3 payloads. ROCKET was modified to incorporate RSAD display variable calculations. These variables are computed in the flight software in order to support RSAD and PDRS status (DISP 16).

### ROCKET SRMS Sim-to-Sim Validation with the digital SPA and POHS

The sim-to-sim validation subroutines were updated along with POHS to have the capability for 5 different payloads for new sim-to-sim validation runs. The masses of the five payloads are

unloaded, 32K, 65K, 180K and 586K lbs. The sim-to-sim validation process with the digital SPA and POHS upgrades to ROCKET was documented in References 15 and 16. The sim-to-sim thruster firing runs comparison of the ASAD results with ROCKET revealed a RMS brake slip problem in ROCKET for the heavier payloads. This was found to be a formatting problem with the thruster firing force/torque input data file for ROCKET for the heavy payloads, and was corrected. Initial grading of ROCKET was completed for the sim-to-sim validation with an overall grade of 84%. SOMBAT simulation input data and validation runs were reviewed and compared to ROCKET for any differences. ROCKET POR calculations incorporating flexible body effects were also reviewed in an effort to improve the validation grade. With motor speed threshold corrections to the Digital SPA implementation, the sim-to-sim validation grade improved from 84% to 87%. The sim-to-sim validation effort was put on hold pending a review of SRMS math models and validation criteria by the SRMS Math Model Working Group.

#### Compensation Curve Update

Compensation testing was performed for all payload sizes and new compensation curves were generated and added to ROCKET. This effort is documented in Reference 16 under 2BODY Compensation Testing Modifications.

#### Implementation of Artificial Damping and Force Filtering into ROCKET

Artificial damping and force filtering algorithms were developed to be used in place of mass factors to stabilize hardware-in-the-loop runs. These techniques were tested in TCDS and demonstrated improved performance. These same algorithms were then tested in the 6DOF HWIL facility to verify the improved performance. They do improve stability, but can lead to higher loads. This effort was documented in Reference 16.

A study of improving simulation stability by using higher order integration methods for integrating the modal equations was also made. This is documented in Reference 16

#### CBM Multi-Stage Capture Testing

Support was provided in the preparation of the 6DOF facility and the ROCKET software for the Boeing RTL and Multi-stage capture tests. Support was provided in the test conduct of these tests. The Direct Drive mode of operation was updated in ROCKET for these tests. This update is documented in Reference 16.

#### Support 6DOF Testing with Astronauts and Trainers

Support was provided for several visits by the astronauts and trainers as they used the 6DOF HWIL facility. This support included testing and verification of PITL operations with the 3A and 5A crew trainers.

#### ROCKET Post Processing Update

ROCKET post processing software was updated to include single joint and direct drive operations. Past post processing problems were identified as due to timing over runs. Post Pro is currently working correctly when there are no process timing problems.

#### SSRMS Math Model

Review of a subset of the 1993 SSRMS data from SPAR was done. There is enough data available to build an initial dynamics model of the SSRMS with no control system. An inverse kinematics algorithm to simplify simulation initialization was not found in any of the SSRMS documents we presently have.

Support was provided in teleconference meetings with JSC and CSA about obtaining the SSRMS model for the MSFC 6DOF facility.

## **2.2 Task B-1: Flight Robotics Laboratory / DOTS Support**

### DOTS Software Upgrades

The frequency based scheduler version of DOTS is currently running successfully with modified controller gains and hardware in the loop. bd Systems integrated the Script File Interpreter, Video Guidance Sensor data collections, and Solar Simulator drivers into the DOTS software.

A separate process for data collection has been created and integrated with the frequency based scheduler and shared memory regions.

The DOTS code has been modified to eliminate software lock ups with the frequency based scheduler. Joint space verification tests of the controllers have been completed. The waist controller was tuned up to improve performance. A digital filter was added to smooth out the waist joint tachometer feedback. The script file interpreter has been integrated with the frequency based scheduler and tested. Data recording has been added to the script file interpreter command set.

Coordinated motion validation tests were performed on the DOTS using the script file utility. A series of tip position and orientation command profiles were executed and compared to the system response as computed from the encoder data to verify the script utility.

Video Guidance Sensor (VGS) communications software was incorporated into the DOTS software. An additional region of shared memory was added to the "blackboard" so that the VGS communications program could share data with the DOTS program. The VGS data is stored with the DOTS time tag to facilitate verification with DOTS position information. Slight changes were made to the DOTS position information format. A 2-3-1 Euler angle sequence is now used (same as the VGS data) for the orientation representation.

The Video Guidance Sensor (VGS) communications software was tested using a program that emulates the VGS data stream. The stored data created by the DOTS Keep process was identified to be the same data as sent by the emulator. DOTS motion testing of the VGS was then performed. During the debugging of the VGS and DOTS software, a potentially dangerous problem appeared. Joint runaways occurred after a long period of operation time had occurred. It is suspected that the encoders are overheating after 12 to 14 hours of continuous operation. The increase in sample rates may have disabled the current software safety measures used to detect erroneous encoder readings. It was also observed during preliminary script mode moves used in set up, that the joint response from the bridge exhibited unacceptable overshoot. The controller gain was reduced to minimize the overshoot. The side effect of increased dynamic error was determined not to effect the VGS motion tests conducted. Also, rate limits were set to low values as to minimize acceleration during script mode. Unlike joint space operations, script mode does not limit joint accelerations except for the new RMS command in script mode (it uses joint space to obtain its commanded position). Output data from the VGS and DOTS were recorded in binary form. The outputs were quick checked at the NT workstation located on the

side of the air-bearing floor using MATLAB. M-files were written to post process the output and plot the results. The sensed position from the VGS was stored in both euler angle form and quaternions (raw information).

The integration of RMS kinematics routines into the DOTS software to compute DOTS position and orientation commands in terms of RMS tip position and orientation is complete. Tests verified the ability to command the tip of the DOTS via a RMS command in script mode. The technical memorandum in reference 4 describing the DOTS RMS script commands was generated.

The document "Results of the DOTS Joint Position Response Tests" was completed and delivered to Linda Brewster along with a draft of the latest user's manual. The user's manual describes the script mode of operation but does not go into detail about the frequency based scheduler version of the DOTS software. Also delivered in Microsoft Word format were:

|               |   |
|---------------|---|
| DRAFT.DOC:    | DOTS Software Overview, 10/27/95  |
| SOFTWARE.DOC: | DOTS Software, Version YDOTS  |
| ENCODR2.DOC:  | Encoder Calibration, 1/9/95   |
| SERVO.DOC:    | Update to the Characterization Test Procedure for the<br>DOTS Joint Servo System, July 1994 |

The encoders have been re-set to the zero markings through software offsets. A check of the encoder 'zero' positions was performed. The bridge joint was found to be almost 2.5 feet off from the last time the encoders were calibrated (8/27/97). The shoulder pitch joint also had significant error. The procedure for determining the zero location for the shoulder pitch has changed. The shoulder pitch is positioned so that the arm extend joint (I-beam) is parallel with the air bearing floor. This can be accomplished by using a level. The difference in the marked zero location (11.8 degrees) and the leveled zero location (15.5 degrees) relative to the previous encoder zero setting was found to be 3.7 degrees. The large errors were not observed during testing and were not present in the VGS data. It is uncertain what the true shoulder pitch error was during testing. The encoder coupling for the bridge joint was inspected. The coupling connected to the encoder shaft and the sprocket shaft was loose. The hose clamp employed on the encoder side was free to float.

The bridge encoder scaling and the post processing of data were checked using a laser range finder (DME 2000) provided by NASA. Reference 7 is the bridge encoder test report written by bd Systems. The laser range finder was set up to measure the position of the payload on the end of the DOTS. The manual control (jog panel) was used to position the DOTS at 10 feet increments from the laser range finder. Visual feedback using the DOTS display was used to ensure the correct displacement. Static data from the laser range finder was recorded by hand. DOTS data was recorded using the KEEP process. The largest error recorded was on the order of 0.2 %. The trolley encoder sprocket was readjusted. It was found to be slightly out of adjustment on its chain. It is possible for the encoder wheel to have jumped a few teeth during fast moves of the trolley joint. However, it is not likely that the alignment problem is the sole source of the trolley positioning errors.

Support was given in setting up the necessary coordinate frames for DOTS accuracy and repeatability tests. The data acquisition in DOTS was altered so that both the laser range finder output and the tip position output are in one file (ASCII format).

### **3.0 SUMMARY**

A wide variety of activities were performed for the Marshall Space Flight Center Six Degree of Freedom Motion Facility and Flight Robotics Laboratory. These activities included dynamic analysis, control system design, software and simulation development, hardware / software integration, HWIL simulation verification and test support. As a result of these activities, significant hardware in the loop simulation capability has been developed and upgraded in the aforementioned facilities.



## REFERENCES

1. Real-time Orbital Contact Kinematics Evaluation Tool (ROCKET) Shuttle Remote Manipulator System (SRMS) Flight-to-Sim Validation Test Report, Mark Slone, MSFC, Dr. Pat Tobbe and Andrea Gilchrist, bd Systems, April 1996.
2. Real-time Orbital Contact Kinematics Evaluation Tool (ROCKET) Shuttle Remote Manipulator System (SRMS) Sim-to-Sim Validation Test Report, Mark Slone, MSFC and Dr. Pat Tobbe, bd Systems, February 1997.
3. TH60020A, RMS Math Model Upgrades and Simulation Validation Support, bd Systems, May 20, 1996.
4. TH70039A, Attachment, DCI TM#081097, DOTS RMS Script Commands, Dr. Pat Tobbe, August 10, 1997.
5. TH70044A, Attachment, DCI TM#092997, ROCKET Flight-to-Sim Validation Runs, Dr. Pat Tobbe, September 29, 1997.
6. TH70052A, Attachment 1, DCI TM#101497-1, ROCKET Flight-to-Sim Validation Runs, Dr. Pat Tobbe, October 14, 1997.
7. TH70052A, Attachment 2, Bridge Encoder Test Report, Marlin Williamson, October 7, 1997.
8. TH80008A, Attachment 1, DCI TM#011398-1, 2BODY Code Modifications and Test Runs, Dr. Pat Tobbe, DCI, Mike Ekbundit, bd Systems, January 13, 1998.
9. TH80008A, Attachment 2, DCI TM#011398-2, Deberth Code Modifications and Test Runs, Dr. Pat Tobbe, DCI, Mike Ekbundit, bd Systems, January 13, 1998.
10. TH80008A, Attachment 3, DCI TM#012798-1, 2BODY Real-Time Simulation Integration and Validation, Dr. Pat Tobbe and Jimmy Compton, January 27, 1998.
11. TH80023A, Attachment 1, DCI TM#052398-1, 6DOF Facility Hardware / Software Integration, Dr. Pat Tobbe, May 23, 1998.
12. TH80034A, Attachment, DCI TM#082698-1, Digital Servo Power Amplifier Model, Dr. Pat Tobbe, August 26, 1998.
13. TH80042A, Attachment 1, DCI TM#103198-1, RMS Simulation POHS Upgrade, Dr. Pat Tobbe, October 31, 1998.
14. TH90030A, Attachment 1, DCI TM#020499-1, RMS Simulation CBM Verification Testing, Dr. Pat Tobbe, February 4, 1999.
15. TH90120A, Attachment, DCI TM#052599-1, RMS Simulation Support, Dr. Pat Tobbe, May 25, 1999.
16. TCD20000049A, Attachment, DCI TM#122799-1, RMS Simulation Support, Dr. Pat Tobbe, December 27, 1999.

bd Systems®  
TCD20000103A  
30 April 2000

Contract No.  
NAS8-40604  
Final Technical Report

## **APPENDICES**

## Reference 1

Real-time Orbital Contact Kinematics Evaluation Tool (ROCKET)  
Shuttle Remote Manipulator System (SRMS) Flight-to-Sim Validation  
Test Report  
Mark Slone, MSFC, Dr. Pat Tobbe and Andrea Gilchrist, bd Systems  
April 1996

[This NASA report, which was jointly authored by MSFC and bd Systems, is not included in this report. This report summarized the Shuttle RMS flight-to-sim analysis of ROCKET. Twenty nine test cases were executed in ROCKET, comparing the simulation model to flight data in thruster firing, direct drive, single joint and manual control modes. The overall grade of the validation of the simulation to flight data was 85.4%.]

## Reference 2

Real-time Orbital Contact Kinematics Evaluation Tool (ROCKET)  
Shuttle Remote Manipulator System (SRMS) Sim-to-Sim Validation  
Test Report, Mark Slone, MSFC and Dr. Pat Tobbe, bd Systems,  
February 1997

[This NASA report, which was jointly authored by MSFC, bd Systems and Dynamic Concepts, is not included in this report. This report summarized the Shuttle RMS sim-to-sim analysis of ROCKET. Sixteen test cases were executed in ROCKET, comparing the simulation model to SPAR's All Singing/All Dancing (ASAD) simulation results. Eight of the cases modeled a 32K payload and eight modeled a 65K payload. The overall grade of the validation of the simulation to ASAD simulation data was 89.2%.]

### Reference 3

TH60020A  
RMS Math Model Upgrades and Simulation Validation Support  
May 20, 1996

Contract No.: NAS8-40604

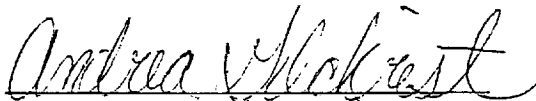
RMS MATH MODEL UPGRADES  
AND  
SIMULATION VALIDATION SUPPORT

20 May 1996

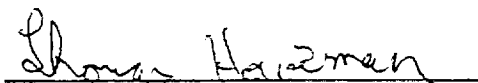
Submitted to:

National Aeronautics and Space Administration  
Simulation System Branch/EB63  
George C. Marshall Space Flight Center  
Marshall Space Flight Center, AL 35812

Prepared by:

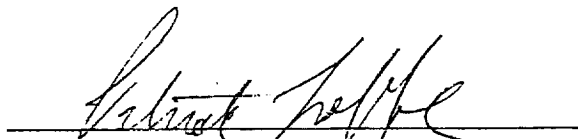


Andrea Gilchrist  
Member of Technical Staff



Dr. Thomas G. Howsman  
bd Systems, Inc. Consultant

Program Manager Approval:



Dr. Patrick Tobbe  
Manager, Space Systems Branch



Control Dynamics, Div. of bd Systems, Inc.  
600 Boulevard South, Suite 304  
Huntsville, AL 35802  
Bus: (205) 882-2650 Fax: (205) 882-2683

## TABLE OF CONTENTS

|            |   |           |
|------------|---|-----------|
| <b>1.0</b> | <b>INTRODUCTION</b>                               | <b>1</b>  |
| <b>2.0</b> | <b>RMS MATH MODEL UPGRADES</b>                    | <b>2</b>  |
| 2.1        | Encoder Quantization Model                        | 2         |
| 2.2        | Tachometer Quantization Model                     | 3         |
| 2.3        | Servo Electronics State Space Model               | 4         |
| 2.4        | Orbiter Thruster Firing Inputs                    | 6         |
| 2.5        | Joint/Motor Friction Models                       | 7         |
| 2.6        | Joint Locking Springs                             | 9         |
| 2.7        | Modified Mode Acceleration Method                 | 11        |
| 2.8        | Virtual Work from Contact Loads                   | 18        |
| 2.9        | Relative Motion Equations                         | 22        |
| <b>3.0</b> | <b>VALIDATION SUPPORT</b>                         | <b>25</b> |
| 3.1        | Strain Gage Location Loads                        | 25        |
| 3.1.1      | Internal Force/Moment Equations                   | 26        |
| 3.1.2      | Algorithm / Software Description                  | 31        |
| 3.1.3      | Constraint Modes                                  | 32        |
| 3.1.4      | Verification Cases                                | 37        |
| 3.2        | RMS Operation Termination Conditions              | 44        |
| 3.3        | Validation Mode Example Runs: Parallel vs. Flight | 45        |
| <b>4.0</b> | <b>CONCLUSIONS/RECOMMENDATIONS</b>                | <b>46</b> |
|            | Appendix A: P20 Thruster Firing Validation Case   | 47        |
|            | Appendix B: D7 Direct Drive Validation Case       | 51        |
|            | Appendix C: S7 Single Joint Validation Case       | 62        |
|            | Appendix D: M5 Man-in-the-loop Validation Case    | 73        |

## NOMENCLATURE

|      |   |  |
|------|---|--|
| ACDS | = | Analytical Contact Dynamics Simulation |
| CBM  | = | Common Berthing Mechanism              |
| CDy  | = | Control Dynamics                       |
| DOF  | = | Degrees-of-Freedom                     |
| EOM  | = | Equations-of-Motion                    |
| GPC  | = | General Purpose Computer               |
| HITL | = | Hardware-in-the-Loop                   |
| JSC  | = | Johnson Space Flight Center            |
| MCIU | = | Manipulator Control Interface Unit     |
| MITL | = | Man-in-the-Loop                        |
| MMAM | = | Modified Mode Acceleration Method      |
| MSFC | = | Marshall Space Flight Center           |
| POR  | = | Point of Resolution                    |
| RCS  | = | Reaction Control System                |
| RMS  | = | Remote Manipulator System              |
| CMS  | = | Component Mode Synthesis               |



## 1.0 INTRODUCTION

Control Dynamics (CDy), a division of bd Systems, under contract to the Marshall Space Flight Center (MSFC), developed a real-time Remote Manipulator System (RMS) simulation for use in Space Station Freedom assembly studies. The RMS simulation is an integral part of MSFC's 6-DOF motion simulator. The 6-DOF simulator's primary purpose is the study of space vehicle contact-dynamics during orbital docking and berthing. This RMS simulation was later modified by CDy in support of Common Berthing Mechanism (CBM) hardware-in-the-loop (HITL) development tests.

The purpose of this document is to present enhancements made to the simulation in support of the formal RMS math model validation process. The most significant upgrade to the model is the addition of the modified mode acceleration method to improve the convergence of the solution with fewer system modes. The free-free substructure modes which describe the flexibility of the RMS booms were replaced with a user defined number of constraint / fixed interface modes. Other changes to the model include sensor quantization effects, servo electronics state space representation, and orbiter thruster firing inputs.

In support of the validation process, software was written to compute internal RMS member loads at the strain gage locations near the shoulder pitch and wrist pitch joints. The RMS software was modified to update the termination conditions for the single joint and direct drive modes of operation. The simulation output routines were also changed to compute data in the proper format for validation. Parameter studies were performed to examine the effects of the number of system modes on the accuracy of the system response.

Section 2.0 of this report documents the upgrades to the math model. The strain gage force/moment algorithm and software is described in Section 3.0. Conclusions and recommendations are made in Section 4.0.

## 2.0 RMS MATH MODEL UPGRADES

This section of the report describes upgrades and enhancements to elements of the RMS simulation. Changes to the joint encoder and motor shaft tachometer sensors to include the quantization effects will be documented. The addition of orbiter thruster firing inputs to the system forcing function, the servo electronics state space model, and joint locking springs will be presented. A modified mode acceleration method (MMAM) including its software implementation will be discussed. Finally, the generalized force terms from the contact loads and mechanism relative motion equations will be described.

### 2.1 Encoder Quantization Model

An electro-optical encoder is fitted to the output shaft of the gear box of each joint. The encoder is used to measure angular position of the joint with an LED light source and a single disk. The output of the encoder is connected to the Manipulator Control Interface Unit (MCIU), a 16 bit fixed point device. The encoder data is relayed from the MCIU to the General Purpose Computer (GPC) which runs the RMS control software. The GPC is a floating point machine. There is a quantization or discretization effect in converting the encoder data from a fixed point format to a floating point format. This effect is modeled in the RMS FORTRAN simulation for the  $i^{\text{th}}$  joint as follows:

```
SUBROUTINE INTFAC
```

```
  I_ANGLE_ENCODE(I) = INT(ENCGAIN*XCUM(I))  
  J_ANGLE_ENCODE(I) = FLOAT(I_ANGLE_ENCODE(I))
```

```
SUBROUTINE KDG
```

```
  JOINT_ANGLES(I) = J_ANGLE_ENCODE(I)*ONE80D2P15
```

The variables are defined as follows:

I\_ANGLE\_ENCODE - Local integer array of truncated integer encoder signal from the MCIU in INTFAC.

ENCGAIN - Conversion factor from joint angle in radians to integer counts based on 16 bit MCIU.

XCUM - RMS simulation degree of freedom state vector, including joint angles, defined in 2MSPRO.INC.

J\_ANGLE\_ENCODE - Global real\*8 array, declared in 24CROSS.INC, of the floating point values of the truncated integer array from the MCIU.

JOINT\_ANGLES - Global real\*8 array, declared in 24CROSS.INC, of the joint angles computed in KDG from the truncated integer values of the MCIU.

ONE80D2P15 - Conversion factor defined in KDG from integer counts to joint angle in degrees.

## 2.2 Tachometer Quantization Model

The tachometers, measuring motor shaft angular velocity, are interfaced to the MCIU similar to the encoders. Therefore, there is a quantization effect in transferring the data from the MCIU fixed point format to the GPC floating point format. The tachometer discretization effect is modeled in the RMS simulation software as follows:

```
SUBROUTINE INTFAC
```

```
TACH_OP(I) = INT( MOT_SPEED(I) * TACGAIN )
```

```
SUBROUTINE KDG
```

```
MOT_RATE_UNFILT(I) = FLOAT( TACH_OP(I) * NINETYD2P10 )
```

The variables are defined as follows:

TACH\_OP - Global integer array, defined in 24CROSS.INC, of the truncated integer tachometer signals from the MCIU.

TACGAIN - Global real\*8 conversion factor from motor shaft rate in rad/sec to integer counts based on the 16 bit MCIU defined in 2MSPRO.INC.

MOT\_SPEED - Global real\*8 array of actual motor shaft rates defined in 2MSPRO . INC.

MOT\_RATE\_UNFILT - Local real\*8 array of motor shaft rates computed in KDG from the truncated integer values of the MCIU.

NINETYD2P10 - Conversion factor defined in KDG from integer counts to angular velocity in rad/sec.

### 2.3 Servo Electronics State Space Model

The models of the electrical components of the RMS joint servo include a low pass filter and a two stage filter in the proportional and velocity paths of the PIV controller. The differential equations for these filters from the original JSC software do not agree with equations derived using conventional state space variable techniques. The following filter equations were derived and implemented into the subroutine SERVO. These equations were later verified with an updated servo routine from JSC. The Laplace transform of the low pass filter is

$$\frac{CMP(s)}{X_D(s)} = \frac{1}{1 + \tau_f s} \quad (2.1)$$

This can be written in the time domain for zero initial conditions as

$$\dot{CMP}(t) = \frac{1}{\tau_f} (-CMP(t) + X_D(t)) \quad (2.2)$$

where

$X_D$  - Digital tachometer input to the filter

$CMP$  - Filter output

$\tau_f$  - Filter time constant

Using rectangular integration, this is implemented in the RMS FORTRAN simulation as

SUBROUTINE SERVO

$$CMP(J) = CMP(J) + (XD(J) - CMP(J)) * DT/TAUF$$

The variables are defined as follows:

CMP - Local real\*8 array of low pass filter output values.

XD - Local real\*8 array of low pass filter tachometer input values.

DT - Global real\*8 scalar, declared in TIME.INC, integration step size.

TAUF - Local real\*8 scalar, filter time constant.

The Laplace transform of the two stage filter shown in Figure 1.6-1 of [1] is

$$\frac{T1(s)}{MS(s)} = \frac{K_1 \tau_1 s}{1 + \tau_1 s} \quad (2.3)$$

$$\frac{ATCH(s)}{T1(s)} = \frac{s}{1 + \tau_2 s} \quad (2.4)$$

$MS$  is the analog motor shaft speed.  $T1$  is the first stage filter output.  $K1$  is the filter gain and  $\tau_1$  is the first stage time constant.  $ATCH$ , the second stage output, is a differential tachometer signal of motor shaft acceleration.  $\tau_2$  is the second stage time constant. These filters can be described in the time domain as

$$\dot{T1}(t) = K1 \dot{MS}(t) - \frac{1}{\tau_1} T1(t) \quad (2.5)$$

$$\dot{ATCH}(t) = \frac{1}{\tau_2} [\dot{T1}(t) - ATCH(t)] \quad (2.6)$$

These equations were integrated into the RMS simulation as follows:

SUBROUTINE SERVO

$$T1DOT(J) = K1 * MOT\_ACC(J-1) - T1(J) / TAU1$$
$$T1(J) = T1(J) + T1DOT(J) * DT$$
$$ATACH(J) = (T1DOT(J) - ATACH(J)) * DT / TAU2 + ATACH(J)$$

T1 - Local real\*8 array of first stage filter outputs

T1DOT - Local real\*8 array of time derivative of first stage filter outputs

MOT\_ACC - Global real\*8 array, defined in 2MSPRO.INC, of motor shaft accelerations

K1 - Global real\*8 scalar, defined in CONSTANT.INC, two stage filter gain

TAU1 - Local real\*8 scalar, first stage filter time constant

ATACH - Local real\*8 array of second stage filter outputs

TAU2 - Global real\*8 scalar, second stage filter time constant

## 2.4 Orbiter Thruster Firing Inputs

For certain math model validation test cases, the Orbiter Reaction Control System (RCS) jets are fired to excite the manipulator. The firing profiles for each thruster are given in files with the flight data to be used in the validation runs. Given the thruster on / off times from the files and the force and moment produced by each thruster, a net force and moment about the orbiter center of mass can be computed as a function of time offline. These time histories can then be read into the RMS simulation for validation runs and applied to the orbiter.

Mark Slone of MSFC generated the time histories of the net loads to be applied to the center of mass of the orbiter during the validation runs. He also modified the simulation to read these files and place the data into global variables which can be used by the dynamics module.

The time histories of the net force and moment about the orbiter center of mass are

read into the global array `F_OLD(1:6, time)`. The first three elements of `F_OLD` are the x, y, and z components of the force vector expressed in the orbiter structural reference frame coordinates. The next three elements are the x, y, and z components of the moment vector expressed in the same coordinate frame. The net forces and moments in `F_OLD` are then loaded into the first six elements of the global array `FLDSP` in the subroutine `RMSPLANT`. These components of `FLDSP` correspond with the translational and rotational degrees of freedom of the base vehicle.

## 2.5 Joint/Motor Friction Models

Frictional losses for the RMS joints are due primarily to the bearings between the RMS members. The friction model for the joints is simple coulomb friction shown in Figure 2-1. Normally, the friction torque has a constant amplitude and opposes the direction of the joint velocity. To avoid a discontinuity at zero velocity, the friction torque is modeled as a straight line with a steep slope in this region.

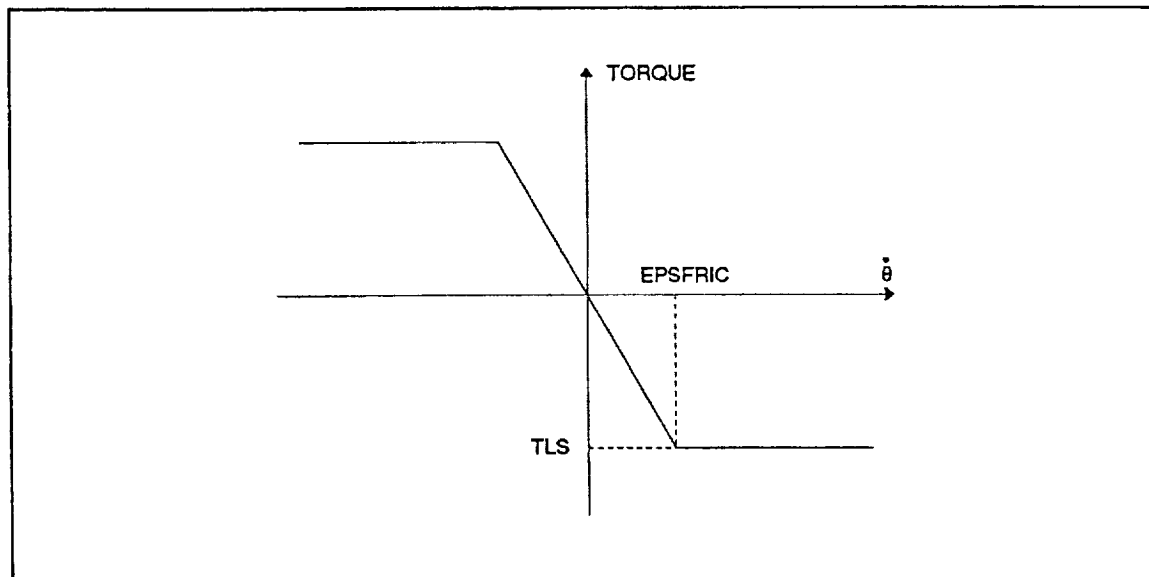


Figure 2.1: Joint Coulomb Friction Model

The joint friction torque is computed in the subroutine `RMSPLANT` as follows:

SUBROUTINE RMSPLANT

JFRICTOR(I) = -TLS(I) \* JOI\_SPEED(I) / (DABS(JOI\_SPEED(I) + EPSFRIC))

where

JFRICTOR - Local real\*8 array of joint friction torques

TLS - Global real\*8 array, declared in 2MSPRO.INC, of nominal running friction levels on joints.

JOI\_SPEED - Global real\*8 array, declared in 2MSPRO.INC, of joint velocities

EPSFRIC - Global real\*8 scalar, declared in 2MSPRO.INC, zero velocity parameter.

A value of .005 for EPSFRIC was selected to best match the flight data from the validation process. The RMS servo motor friction is also modeled as coulomb friction with the addition of stiction effects. These frictional losses result from the bearings in the motor housing which support the motor output shaft. To incorporate stiction, two motor acceleration arrays are computed with and without friction. The motor accelerations are integrated to yield shaft velocities. If the velocities with and without friction have the same sign, the solution incorporating friction is selected. If the directions differ, the friction torque is large enough to overcome the applied torques from the motor field and gear box reaction. The motor is essentially locked by the friction and the shaft acceleration and velocity are set to zero. As with the RMS joints, the motor friction torque opposes the direction of the shaft velocity. However, if the shaft velocity is zero, the friction torque resists the applied torques from the motor field and gearbox.

The motor friction and acceleration arrays are computed in RMSPLANT as follows:

```
C
C   FORM MOTOR STATE DOT VECTORS WITH AND WITHOUT FRICTION
C
DO I=1,6
IF (MOT_SPEED(I+1).EQ.0.D0) THEN
  IF (JBRK(I+1).GT.0) THEN
    MFRICTOR(I) = -SIGN(TBR(I), MOT_TORQUE(I+1) - MBDTOR(I))
  ELSE
```



```

        MFRICTOR(I) = -SIGN(TMST,MOT_TORQUE(I+1)-MBDTOR(I))
    ENDIF
ELSE
    IF (JBRAK(I+1).GT.0) THEN
        MFRICTOR(I) = -SIGN(TBR(I),MOT_SPEED(I+1))
    ELSE
        MFRICTOR(I) = -SIGN(TMST,MOT_SPEED(I+1))
    ENDIF
ENDIF
TEMP_MOT_ACC(I) = (MOT_TORQUE(I+1) + MFRICTOR(I)
1              - MBDTOR(I)) / JMGR(I)
MOT_ACC(I) = (MOT_TORQUE(I+1) - MBDTOR(I)) / JMGR(I)
ENDDO

```

where

MOT\_SPEED - Global real\*8 array, declared in 2MSPRO.INC, of motor shaft velocities

JBRAK - Global integer\*4 array, declared in 2MSPRO.INC, of motor brake flags. Brake is engaged for a value of 1.

MFRICTOR - Local real\*8 array of motor friction torques

TBR - Global real\*8 array, declared in 2MSPRO.INC, of motor brake friction torques

TMST - Global real\*8 array, declared in 2MSPRO.INC, of motor running friction torques

MOT\_TORQUE - Global real\*8 array, declared in 2MSPRO.INC, of motor field torques

MBDTOR - Global real\*8 array, declared in 2MSPRO.INC, of gear box reaction torques on motor shafts.

JMGR - Global real\*8 array, declared in 2MSPRO.INC, of motor shaft inertias

TEMP\_MOT\_ACC - Global real\*8 array, declared in 2MSPRO.INC, of motor shaft accelerations including friction

MOT\_ACC - Global real\*8 array, declared in 2MSPRO.INC, of motor shaft accelerations without friction

## 2.6 Joint Locking Springs

In order to increase the effects of modal damping on the higher frequency system

modes, a set of torsional locking springs were attached to the actively controlled joints in the structural model of the RMS. The use of these springs effectively lowered the system natural frequencies and increased the number of modes retained in the mode displacement solution.

The restoring torque from these springs is added to the right hand side of the equations of motion to effectively "unlock" these joints. The locking springs were added to the stiffness matrix of the structural model in the subroutine LDASPIN through the 3 x 3 stiffness matrices defined by the variable RKLR. The locking spring stiffness variable used by RKLR is global real\*8 scalar KLOCK declared in CONSTANT.INC. It is assigned a value of  $1 \times 10^4$  ft-lb/rad in the subroutine DATABLOC.

The restoring torque is computed as the product of the locking spring constant and the change in the joint angle from its value used to compute the stiffness matrix in the subroutine LDASP. The joint angles used in the re-linearization process are computed in the subroutine RELIN using modal coordinates as

```
SUBROUTINE RELIN
```

```
XEQU(I) = XEQU(I) + PHIM(I,J)*ETA_REF(J)
```

where

XEQU - Global real\*8 array, declared in 2MSPRO.INC, of joint angles at relin time used by subroutine LDASP

PHIM - Global real\*8 matrix, declared in 22CROSS.INC, of system mode shapes.

ETA\_REF - Global real\*8 array, declared in 22CROSS.INC, of reference modal coordinates.

The restoring torque is computed in subroutine RMSPLANT and added to the joint gear box torque as

```
SUBROUTINE RMSPLANT
```

```
X_JOINT_TEMP(I) = XCUM(I) - XEQU(I)
```

$$TGJF(I) = TGJF(I) + KLOCK * X\_JOINT\_TEMP(I) / J\_GEAR\_RATIO(I)$$

where

X\_JOINT\_TEMP - Local real\*8 array of joint motion relative to relinearization configuration.

J\_GEAR\_RATIO - Global real\*8 array, declared in CONSTANT.INC, of joint gear ratios.

TGJF - Local real\*8 array of joint torques from gear box flexibility, locking springs, and joint friction.

## 2.7 Modified Mode Acceleration Method

The RMS dynamics model used in this simulation is based on the mode displacement method. As previously documented, the physical variable  $\mathbf{q}$  of the RMS model are computed as

$$\mathbf{q} = \mathbf{q}_0 + \dot{\mathbf{q}}_0 t + \delta \quad (2.7)$$

$\mathbf{q}$  is an array composed of base vehicle translations and rotations, RMS joint angles, and generalized time coordinates for the substructure bending modes which describe the deformation of the elastic members.  $\mathbf{q}_0$  and  $\dot{\mathbf{q}}_0$  are the initial position and rates of  $\mathbf{q}$ . A set of system bending modes  $\phi$  is computed for the base vehicle, RMS, and payload assembly from the mass and stiffness matrices generated by LDASP.

$\delta$  is the perturbation or change of  $\mathbf{q}$  from  $\mathbf{q}_0$  due to system forcing functions. At each relinearization cycle,  $\mathbf{q}_0$ ,  $\dot{\mathbf{q}}_0$ , and  $\delta$  are reset as follows:

$$\begin{aligned} \mathbf{q}_0 &= \mathbf{q} \\ \dot{\mathbf{q}}_0 &= \dot{\mathbf{q}} \\ \delta &= 0 \end{aligned} \quad (2.8)$$

The transformation to modal coordinates is accomplished through

$$\mathbf{q} = \Phi(\eta_0 + \dot{\eta}_0 t + \eta) \quad (2.9)$$

When all of the system modes are used, this is an exact equation. However, this is an approximation when a subset of  $\Phi$  is used in Eq. (2.9).

The modal coordinates are computed at initialization and at each re-linearization time as

$$\begin{pmatrix} \eta_0 \\ \dot{\eta}_0 \\ \eta \end{pmatrix} = \Phi^T m \begin{pmatrix} \mathbf{q}_0 \\ \dot{\mathbf{q}}_0 \\ \delta \end{pmatrix} \quad (2.10)$$

$$(2.11)$$

$$(2.12)$$

If the values of the generalized coordinates computed using a truncated set of modes in Eqs. (2.10) through (2.12) are substituted into Eq. (2.9), there will be a difference between the physical variables  $\mathbf{q}$  generated using Eqs. (2.7) and (2.9). This is the source of the "drift" problem. At each re-linearization time, the modal coordinates are re-initialized for the new modes and cannot exactly recreate the physical coordinates at the time. This is a result of trying to use an m-dimensional vector space to represent an n-dimensional vector where m is less than n.

In order to eliminate the "drift" problem due to modal truncation, a modified mode acceleration method (MMAM) was developed by Control Dynamics to approximate the response of the truncated higher frequency system modes.

The system modal acceleration equation for the  $i^{\text{th}}$  mode is

$$\ddot{\eta}_i = \Phi_i^T \mathbf{Q} - 2\zeta\omega_i \dot{\eta}_i - \omega_i^2(\eta_i + \dot{\eta}_{0,i} t) - 2\zeta\omega_i \dot{\eta}_{0,i} - \omega_i^2(\eta_{0,i} - \eta_{ref,i}) \quad (2.13)$$

where

$\omega_i$  =  $i^{\text{th}}$  System natural frequency

$\zeta$  = Modal damping factor

$\mathbf{Q}$  = Generalized forces and torques

$t$  = relinearization time

Note that  $\eta_{0_i}$ ,  $\dot{\eta}_{0_i}$ , and  $\eta_{ref_i}$  are constants between relinearization cycles.

The static response of the truncated modes due to the applied loads in  $\mathbf{Q}$  can be calculated from Eq. (2.13) by solving for the steady state value of  $(\eta + \dot{\eta}_0 t + \eta_0)$  to obtain

$$\eta_{TD_i} = \frac{\phi_i^T \mathbf{Q}}{\omega_i^2} \quad (2.14)$$

This equation is only valid if  $\mathbf{Q}$  is not a function of  $\eta_i$ . When this occurs, a set of simultaneous equations must be solved for  $\mathbf{Q}$  and  $\eta_i$ .

In order to avoid this problem in the RMS simulation, a low pass filter is implemented to introduce a time delay which essentially uncouples the  $\mathbf{Q}$  and  $\eta$  for discrete points in time. The low pass filter is implemented as follows:

$$\frac{\eta_i}{\eta_{TD_i}} = \frac{1}{s^2 + 2\zeta_f \omega_f s + \omega_f^2} \quad (2.15)$$

where  $s$  is the Laplace operator,  $\omega_f$  is the filter cut off frequency, and  $\zeta_f$  is the filter damping factor.

From Eq. (2.15), the algorithm generates

$$\ddot{\eta}_i = -2\zeta_f \omega_f \dot{\eta}_i - \omega_f^2 (\eta_i + \dot{\eta}_{0_i} t - \eta_{TD_i}) - \ddot{\eta}_{\Delta f_i} \quad (2.16)$$

where

$$\ddot{\eta}_{\Delta f_i} = 2\zeta_f \omega_f \dot{\eta}_{0_i} + \omega_f^2 (\eta_{0_i} - \eta_{ref_i}) \quad (2.17)$$

The modal accelerations from Eq. (2.16) are solved numerically to compute  $\eta$  for the truncated modes. The system response is computed from Eq. (2.9) using all system modes. Note that  $\omega_i$  in Eq. (2.14) is typically higher than  $\omega_f$ , and the integration step size is determined by  $\omega_f$ .

In order to implement the modified mode acceleration into the RMS software,

several subroutines had to be altered. The following section will give a brief description of the subroutines used for the mode acceleration equations in the previous section, a portion of the altered code, and a definition of the variables.

Subroutine LDASP generates the system mass and stiffness matrices,  $AM$  and  $AK$ , based on the current system configuration. The calculation of  $AM$  and  $AK$  is based on the total number of degrees of freedom ( $NDOF$ ).

Subroutine EIGEN1 generates the eigensolution of the following generalized eigen problem:

$$k\phi = \omega^2 m\phi \quad (2.18)$$

where

$m$  = Mass Matrix ( $AM$ )

$k$  = Stiffness Matrix ( $AK$ )

$\omega$  = System Natural Frequency ( $OMEGA$ )

$\phi$  = System Mode Shapes ( $PHIM$ )

The variables in the parentheses indicate the corresponding names used in the RMS code. Subroutine EIGEN1 also determines the number of truncated modes that will be used in the solution. This is done by comparing the magnitude of the computed natural frequencies to the fixed frequency limit denoted by  $FREQ_{LIM}$ . The number of modes below  $FREQ_{LIM}$  take the value of  $NMODE$ . Although  $NMODE$  is computed,  $OMEGA$  and  $PHIM$  are calculated based on the total degrees of freedom,  $NDOF$ , rather than  $NMODE$ .

Subroutine TPHI computes the variables  $PHIMTAM$  and  $ETA\_REF$  used in the calculation of the system modal acceleration in Eq. (2.13). These computations are based on  $NDOF$ . The implementation of these variables into the subroutine TPHI is shown below:

$$PHIMTAM(I,J) = PHIMTAM(I,J) + PHIM(K,I) * AM(K,J)$$

$$ETA\_REF(I) = ETA\_REF(I) + PHIMTAM(I,J) * XREF(J)$$

where

PHIM - Global real\*8 matrix of system eigenvectors ( $\phi$ ) declared in  
22CROSS.INC

AM - Global real\*8 of mass matrix ( $m$ ) declared in 2SECPRO.INC

PHIMTAM - Global real\*8 matrix product of the transpose of PHIM and AM  
( $\phi^T m$ ) declared in 22CROSS.INC

XREF - Global real\*8 system equilibrium configuration ( $q_{ref}$ ) declared in  
22CROSS.INC

ETA\_REF - Global real\*8 generalized coordinates describing the system  
configuration at relinearization time ( $\eta_{ref}$ ) declared in 22CROSS.INC

Subroutine RELIN computes the initial generalized coordinates to be used in the dynamic solution process after each relinearization. All of the variable calculations are based on NDOF rather than NMODE. First, the initial state vector, XINIT, is reset as seen in Eq. (2.8) using the following code:

```
XINIT(I) = XCUM(I)
```

```
XINIT_DOT(I) = XCUM_DOT(I)
```

where

XINIT - Global real\*8 values of system degrees of freedom at relinearization time ( $q_0$ ) declared in 2MSPRO.INC

XINIT\_DOT - Global real\*8 time derivative of XINIT ( $\dot{q}_{init}$ ) declared in  
2MSPRO.INC

XCUM - Global real\*8 cumulative values of system states ( $q$ ) declared in  
2MSPRO.INC

XCUM\_DOT - Global real\*8 time derivative of XCUM ( $\dot{q}$ ) declared in 2MSPRO.INC

Next, the modal coordinates are computed at initialization and at each relinearization time as in Eqs. (2.10) and (2.11). This is represented by the following code:

```
ETA_INIT(I) = ETA_INIT(I) + PHIMTAM(I,J)*XINIT(J)
```

$$\text{ETA\_INIT\_DOT}(I) = \text{ETA\_INIT\_DOT}(I) + \text{PHIMTAM}(I, J) * \text{XINIT\_DOT}(J)$$

where

ETA\_INIT - Global real\*8 generalized coordinates describing the system configuration at relinearization time ( $\eta_0$ ) declared in 2MSPRO.INC

ETA\_INIT\_DOT - Global real\*8 time derivative of ETA\_INIT ( $\dot{\eta}_0$ ) declared in .INC

A portion of the system modal acceleration calculation in Eq. (2.13) for the retained modes is obtained in RELIN based on the value of NMODE as follows:

$$\text{ETA\_DDOT\_DELTA}(I) = \text{ZETA\_OMEGA}(I) * \text{ETA\_INIT\_DOT}(I) + \text{OMEGA\_SQ}(I) * (\text{ETA\_INIT}(I) - \text{ETA\_REF}(I))$$

where

ZETA\_OMEGA - Global real\*8, ( $2\zeta\omega$ ), declared in 2MSPRO.INC

OMEGA\_SQ - Global real\*8 natural frequency squared ( $\omega^2$ ) declared in 2MSPRO.INC

ETA\_DDOT\_DELTA - Global real\*8,  $\ddot{\eta}_\Delta$ , declared in 2MSPRO.INC

The equivalent expression for the filtered modes in Eq. (2.17) is also computed, but its calculation is performed from (NMODE + 1) to NDOF using the following code:

$$\text{ETA\_DDOT\_DELTA} = \text{ZETA\_OMEGA\_FIL} * \text{ETA\_INIT\_DOT}(I) + \text{OMEGA\_FIL\_SQ} * (\text{ETA\_INIT}(I) - \text{ETA\_REF}(I))$$

where

ZETA\_OMEGA\_FIL - Global real\*8,  $2\zeta_f\omega_f$ , defined in 2MSPRO.INC

OMEGA\_FIL\_SQ - Global real\*8 filter cut off frequency squared ( $\omega_f^2$ ) defined in 2MSPRO.INC

Subroutine INTFAC calculates the system configuration states and velocities. This data is used to generate encoder and tachometer signals and relative docking port information. To incorporate mode acceleration, the following code calculating the



perturbation variables was based on NDOF instead of NMODE:

$$XLDSP(I) = XLDSP(I) + PHIM(I,J)*ETA(J)$$

$$XDOTLDSP(I) = XDOTLDSP(I) + PHIM(I,J)*ETA\_DOT(J)$$

$$XDDOT\_LDSP(I) = XDDOT\_LDSP(I) + PHIM(I,J)*ETA\_STAT\_DOT(J)$$

where

XLDSP - Global real\*8 system perturbation away from linearized configuration ( $\delta$ ) declared in 2MSPRO.INC

XDOTLDSP - Global real\*8 time derivative of XLDSP ( $\dot{\delta}$ ) defined in 2MSPRO.INC

XDDOT\\_LDSP - Global real\*8 time derivative of XDOTLDSP ( $\ddot{\delta}$ ) declared in 2MSPRO.INC

ETA - Global real\*8 generalized modal coordinates describing structure flexibility ( $\eta$ ) declared in 2MSPRO.INC

ETA\\_STAT\\_DOT - Global real\*8 time derivative of state vector of generalized modal coordinates declared in 2MSPRO.INC

Another calculation performed for mode acceleration in Eq. (2.1) includes the following code where T2SEC is the relinearization time:

$$XCUM(I) = XINIT(I) + XLDSP(I) + XINIT\_DOT(I)*T2SEC$$

$$XCUM\_DOT(I) = XINIT\_DOT(I) + XDOTLDSP(I)$$

The subroutine RMSPLANT computes the state dot vectors for the motor shaft and system generalized coordinates. A generalized forcing function, FGEN, which drives the equations of motion is calculated within RMSPLANT. FGEN consists of the gear train torque, the joint friction torque, and the contact force/moment contributions. FGEN represents  $\phi^T Q$  in Eq. (2.13), and is calculated based on all degrees of freedom (NDOF).

Using the mode superposition technique, the generalized coordinate state dot

vector, ETA\_STAT\_DOT, is calculated using Eq. (2.13). The value of ETA\_DDOT\_DELTA is computed in the subroutine RELIN. The following code performs the ETA\_STAT\_DOT calculation based on NMODE which is the number of modes selected based on FREQLIM:

```
ETA_STAT_DOT(I) = FGEN(I) - ZETA_OMEGA(I)*ETA_DOT(I) -
                 OMEGA_SQ(I)*(ETA(I) + ETA_INIT_DOT(I)*T2SEC)
                 - ETA_DDOT_DELTA(I)

ETA_STAT_DOT(I+NDFMX) = ETA_STAT(I)
```

where

ETA\_DOT - Global real\*8 time derivative of ETA ( $\dot{\eta}$ ) declared in 2MSPRO.INC

NDFMX - Global integer\*4 scalar of maximum number of total degrees of freedom (flexible and link dof's) declared in CONSTANT.INC

Calculation of ETA\_STAT\_DOT for the mode acceleration type terms was added to RMSPLANT with the following code:

```
ETA_STAT_DOT(I) = -ZETA_OMEGA_FIL*ETA_STAT(I) -
                 OMEGA_FIL_SQ*(ETA_STAT(I+NDFMX) -
                 FGEN/OMEGA_SQ) - ETA_DDOT_DELTA(I)

ETA_STAT_DOT(I+NDFMX) = ETA_STAT(I)
```

## 2.8 Virtual Work from Contact Loads

The generalized forces,  $Q$ , from the equations of motion of Eq. (2.13) resulting from contact loads are best derived from the following virtual work expression:

$$\delta W = F_{D1} \cdot \delta r_{D1} + F_{D2} \cdot \delta r_{D2} + T_{D1} \cdot \delta \theta_{D1} + T_{D2} \cdot \delta \theta_{D2} \quad (2.18)$$

where

$F_{D1}$  - Contact force vector acting on passive  $D1$  frame

$\delta r_{D1}$  - Virtual translation vector of  $D1$

$F_{D2}$  - Contact force vector acting on active  $D2$  frame

$\delta r_{D2}$  - Virtual translation vector of  $D2$

$T_{D1}$  - Contact torque vector acting on  $D1$  frame

$\delta \theta_{D1}$  - Virtual rotation vector of  $D1$

$T_{D2}$  - Contact torque vector acting on  $D2$  frame

$\delta \theta_{D2}$  - Virtual rotation vector of  $D2$

A force and moment vector pair equivalent to that acting at  $D1$  can be computed about  $S1$  as:

$$F_{D1} = F_{S1} \quad (2.19)$$

$$T_{D1} = T_{S1} + r_{S1D1} \times F_{S1} \quad (2.20)$$

where

$F_{S1}$  - Contact force vector acting at  $S1$

$T_{S1}$  - Contact torque vector acting about  $S1$

$r_{S1D1}$  - Position vector of  $S1$  with respect to  $D1$

The contact forces and moments acting at  $D2$  can be expressed in terms of those at  $D1$  by assuming equal and opposite reactions.

$$F_{D2} = -F_{D1} \quad (2.21)$$

$$T_{D2} = -T_{D1} - r_{D1D2} \times F_{D1} \quad (2.22)$$

where

$r_{D1D2}$  - Position vector of  $D1$  with respect to  $D2$

Substituting Eqs. (2.19) - (2.22) into Eq. (2.18) yields

$$\delta W = F_{D1} \cdot (\delta r_{D1} - \delta r_{D2}) + T_{D1} \cdot (\delta \theta_{D1} - \delta \theta_{D2}) - (r_{D1D2} \times F_{D1}) \cdot \delta \theta_{D2} \quad (2.23)$$

Writing the relative motion terms as

$$\delta \theta_{D2/D1} = \delta \theta_{D2} - \delta \theta_{D1} \quad (2.24)$$

$$\delta r_{D2/D1} = \delta r_{D2} - \delta r_{D1} - (\delta \theta_{D1} + \delta \theta_{D2/D1}) \times r_{D2D1} \quad (2.25)$$

and substituting into Eq. (2.23) using vector triple products produces

$$\delta W = -F_{D1} \cdot \delta r_{D2/D1} - T_{D1} \cdot \delta \theta_{D2/D1} \quad (2.26)$$

The relative motion between the berthing ports is written in terms of RMS system variables in inertial coordinates as

$$\delta r'_{D2/D1} = \sum_{j=1}^n \phi'_{D2/D1_j} \delta q_j \quad (2.27)$$

$$\delta \theta'_{D2/D1} = \sum_{j=1}^n \psi'_{D2/D1_j} \delta q_j \quad (2.28)$$

where

$\phi'_{D2/D1_j}$  - Translation vector of  $D2$  relative to  $D1$  due to motion of  $q_j$

$\psi'_{D2/D1_j}$  - Rotation vector of  $D2$  relative to  $D1$  due to motion of  $q_j$

The generalized force vector for the  $j^{\text{th}}$  RMS DOF expressed in terms of force/moment

sensor output is

$$Q_j = -[IS1]F_{S1}^{S1} \cdot \phi'_{D2/D1_j} - ([IS1]T_{S1}^{S1} + [IS1](r_{S1D1}^{S1} \times F_{S1}^{S1})) \cdot \psi'_{D2/D1_j} \quad (2.29)$$

where

[IS1] - Transformation matrix from S1 to inertial coordinates

The generalized forces from mechanism contact are computed in the simulation in the subroutine RMSPLANT as follows:

SUBROUTINE RMSPLANT

```

C
C   MOVE CONTACT FORCES AND MOMENTS FROM SENSOR LOCATION TO C C
C   BASE VEHICLE DOCKING PORT
C
  TD1S1(1) = TS1S1(1) + RS1D1S1(2)*FS1S1(3) -
             RS1D1S1(3)*FS1S1(2)
  TD1S1(2) = TS1S1(2) + RS1D1S1(3)*FS1S1(1) -
             RS1D1S1(1)*FS1S1(3)
  TD1S1(3) = TS1S1(3) + RS1D1S1(1)*FS1S1(2) -
             RS1D1S1(2)*FS1S1(1)
C
C   TRANFORM CONTACT FORCES AND MOMENTS FROM SENSOR FRAME
C   TO INERTIAL FRAME
C
  FD1I(1) = IS1(1,1)*FS1S1(1) + IS1(1,2)*FS1S1(2) +
&          IS1(1,3)*FS1S1(3)
  FD1I(2) = IS1(2,1)*FS1S1(1) + IS1(2,2)*FS1S1(2) +
&          IS1(2,3)*FS1S1(3)
  FD1I(3) = IS1(3,1)*FS1S1(1) + IS1(3,2)*FS1S1(2) +
&          IS1(3,3)*FS1S1(3)
  TD1I(1) = IS1(1,1)*TD1S1(1) + IS1(1,2)*TD1S1(2) +
&          IS1(1,3)*TD1S1(3)
  TD1I(2) = IS1(2,1)*TD1S1(1) + IS1(2,2)*TD1S1(2) +
&          IS1(2,3)*TD1S1(3)
  TD1I(3) = IS1(3,1)*TD1S1(1) + IS1(3,2)*TD1S1(2) +
&          IS1(3,3)*TD1S1(3)

  ENDIF

```

```
DO I = 1,NDOF
```

```
    FLDSP(I) = FLDSP(I) - FD1I(1) * PHL(17,I,1) - FD1I(2) *  
                PHL(17,I,2) - FD1I(3) * PHL(17,I,3) - TD1I(1) *  
                PSL(17,I,1) - TD1I(2) * PSL(17,I,2) - TD1I(3) *  
                PSL(17,I,3)
```

```
ENDDO
```

where

TD1S1 - Local real\*8 vector of contact moments about D1 in S1 coordinates

TS1S1 - Global real\*8 vector, declared in 2MSPRO.INC, of contact moments about S1 in S1 coordinates

RS1D1S1 - Global real\*8 vector, declared in 2MSPRO.INC, locating S1 with respect to D1 in S1 coordinates

FD1I - Local real\*8 vector of contact forces acting at D1 in inertial coordinates

TD1I - Local real\*8 vector of contact torques acting at D1 in inertial coordinates

FLDSP - Global real\*8 array, declared in 2MSPRO.INC, of generalized force components

PHL - Global real\*8 matrix, declared in 22CROSS.INC, of translational perturbation gains

PSL - Global real\*8 matrix, declared in 22CROSS.INC, of rotational perturbation gains

## 2.9 Relative Motion Equation

The output of the RMS math model is used by the SIXDOF, COMP, and LEGLEN subroutines to compute leg length commands. The resulting table motion tracks the simulated relative motion between the berthing ports D1 and D2. The RMS simulation computes the relative translation and orientation between the berthing ports as follows in the subroutine INTFAC:

$$\dot{R}_{D2D1}^{D1} = [D1I] \phi_{D2/D1} \dot{q} \quad (2.30)$$

$$(\omega_{D2} - \omega_{D1})' = \omega_{D2/D1}' = \psi_{D2/D1} \dot{q} \quad (2.31)$$

where

$\dot{R}_{D2D1}^{D1}$  - Inertial time derivative of position of  $D2$  with respect to  $D1$  in  $D1$  coordinates

$[D2I]$  - Transformation matrix from inertial to  $D2$  coordinates

$\omega_{D2/D1}'$  - Angular velocity of  $D2$  relative to  $D1$  in inertial coordinates

$$[D1D2] = [D1D2''] [D2''D2'] [D2'D2] \quad (2.32)$$

where

$[D1D2]$  - Transformation matrix from  $D2$  to  $D1$  coordinates

$[D1D2'']$  - Fixed 180° rotation about  $z$  axis

$D2''$  is a coordinate frame fixed to  $D1$  but parallel to  $D2$  when the mechanism misalignment is zero.

$[D2'D2]$  - Transformation matrix between  $D2''$  and  $D2$  at relinearization time.  
This is a constant transformation between relin times.

$$[D2''D2'] = 1 - \sin\theta_m \tilde{u} + (1 - \cos\theta_m) \tilde{u}\tilde{u} \quad (2.33)$$

where

$\theta_m$  - Perturbation angle between  $D2'$  and  $D2''$

$$u = \Delta\theta_{D2/D1}' / \theta_m \quad (2.34)$$

$$\Delta \theta'_{D2/D1} = \psi_{D2/D1} (\phi \eta + \dot{q}_0 (t - t_{relin})) \quad (2.35)$$

where

$u$  - Unit vector along perturbation rotation axis

$t$  - Current simulation time

$t_{relin}$  - Time of last relinearization

The derivative of  $R_{D2D1}$  with respect to  $D1$  in  $D1$  coordinates is written as

$$\overset{\circ}{R}_{D2D1}^{D1} = [D1I] (\dot{R}_{D2D1}' - \omega_{D1}' \times R_{D2D1}') \quad (2.36)$$

where

$\omega_{D1}'$  - Angular velocity vector of  $D1$  in inertial coordinates

$R_{D2D1}'$  - Position of  $D2$  with respect to  $D1$  in inertial coordinates

$$R_{D2D1}' = R_{D2D1}' + \phi_{D2/D1} (q - q_0) \quad (2.37)$$

where

$R_{D2D1}' - R_{D2D1}'$  at relin time



### 3.0 VALIDATION SUPPORT

In order to support the formal validation of the RMS model, Control Dynamics developed algorithms and software to calculate internal member loads for the manipulator. Control Dynamics constructed NASTRAN models of the RMS members and computed component modes based on a modified Craig-Bampton technique. The RMS flight software was modified to implement validation termination conditions and compute point of resolution (POR) output data.

#### 3.1 Strain Gage Location Loads

Links three and four of the RMS have been instrumented with strain gages at the locations indicated in Figure 3.1. Strain gage data was obtained during several early STS flights. The actual strain data has been processed into what is labeled as cross-sectional load data. The load data represents the internal loads existing at the strain gage locations. This section develops the equation of the forces and moments present at the strain gage locations, discusses the software algorithm, and describes the static and dynamic tests performed for verification.

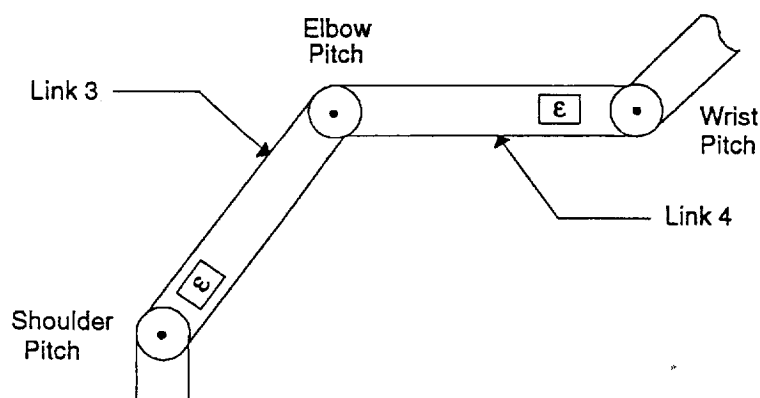


Figure 3.1: Strain Gage Locations on Links Three and Four

### 3.1.1 Internal Force/Moment Equations

In order to compute the force and moment equations for links three and four, each link had to be divided into two substructures using the strain gage locations as dividing points. The substructure equations of motion used to compute the interface loads can be written as

$$M\ddot{x} + C\dot{x} + Kx = F \quad (3.1)$$

The matrices from the equation above can be partitioned as follows:

$$\begin{bmatrix} M_{ii} & M_{io} \\ M_{oi} & M_{oo} \end{bmatrix} \begin{pmatrix} \ddot{x}_i \\ \ddot{x}_o \end{pmatrix} + \begin{bmatrix} C_{ii} & C_{io} \\ C_{oi} & C_{oo} \end{bmatrix} \begin{pmatrix} \dot{x}_i \\ \dot{x}_o \end{pmatrix} + \begin{bmatrix} K_{ii} & K_{io} \\ K_{oi} & K_{oo} \end{bmatrix} \begin{pmatrix} x_i \\ x_o \end{pmatrix} = \begin{pmatrix} F_i \\ F_o \end{pmatrix} \quad (3.2)$$

The subscript 'i' denotes the strain gage interface values while the 'o' denotes all other values. Eq. (3.2) is the equation of motion of a component model that has been cut at the strain gage location. The  $x_o$  are considered to be generalized degrees of freedom, some or all of which may represent physical degrees of freedom. By considering the interface contributions of the Eq. (3.2), the following simplified equation can be derived:

$$F_i = M_i \ddot{x}_i + C_i \dot{x}_i + K_i x_i \quad (3.3)$$

where  $M_i$ ,  $C_i$ , and  $K_i$  are the upper six rows of the mass, damping, and stiffness matrices in Eq. (3.2), respectively. The interface force,  $F_i$ , may have both external and internal loads to consider. The values of  $x_i$ ,  $\dot{x}_i$ , and  $\ddot{x}_i$  can be a linear combination of generalized model vectors, i.e.,

$$x_i = \phi \eta \quad (3.4)$$

where both rigid body and flexible modes can be present in the  $\phi$  set.

In order to extract the mass, damping, and stiffness matrices in Eq. (3.2), a finite element (NASTRAN) model was constructed. The listing for both links 3 and link 4 is shown below:

```

$
ID HOWSMAN,TG
SOL,63 $ V65 - MODE SHAPES
TIME 9999
CEND
TITLE = RMS LINK 3 FEM WITH KINK (L3C_BEAM.DAT)
SUBTITLE=MODAL SOLUTION (BEAM)
ECHO=SORT
METHOD=10
$
$SPC=11
$
DISP = ALL
SEALL = ALL
$
$
BEGIN BULK
$
$
PARAM COUPMASS 1
PARAM AUTOSPC YES
PARAM GRDPNT 0
PARAM,USETPRT,0
PARAM,NEWSEQ,-1
$
EIGR,10,MGIV,,1500.
$
$11111111222222223333333344444444555555556666666677777777888888889999999900000000
GRID      1  0  0.0  0.0  0.0  0
GRID      2  0  1.169  0.0  0.0  0
GRID      3  0  2.372  0.0  0.0  0
GRID      4  0  6.478  0.0  0.0  0
GRID      5  0 10.584  0.0  0.0  0
GRID      6  0 14.690  0.0  0.0  0
GRID      7  0 18.796  0.0  0.0  0
GRID      8  0 20.921  0.0  -5  0
$234567890123456789012345678901234567890123456789012345678901234567890
CBEAM      1  1  1  2  1.0  1.0  0.0
CBEAM      2  1  2  3  1.0  1.0  0.0
CBEAM      3  2  3  4  1.0  1.0  0.0
CBEAM      4  2  4  5  1.0  1.0  0.0
CBEAM      5  2  5  6  1.0  1.0  0.0
CBEAM      6  2  6  7  1.0  1.0  0.0
CBEAM      7  3  7  8  1.0  1.0  0.0
$234567890123456789012345678901234567890123456789012345678901234567890
PBEAM      1  101 0.345 .362E-2 .392E-2 0.0 .476E-2
PBEAM      2  102 .198E-1 .709E-2 .709E-2 0.0 .938E-2
PBEAM      3  103 0.385 .217E-2 .262E-2 0.0 .336E-2
MAT1      101 .144E10 .56E9 5.26
MAT1      102 .144E10 .56E9 2.9
MAT1      103 .144E10 .56E9 5.26
$
ENDDATA

```

---

\$  
ID HOWSMAN,TG  
SOL 63  
TIME 100  
CEND  
TITLE = RMS LINK 4 FEM WITH KINK (LINK\_4C.DAT)  
SUBTITLE = MODAL SOLUTION  
ECHO = UNSORT  
METHOD = 10

\$  
\$ SPC = 11

\$  
DISP = ALL  
SEAL = ALL

\$  
BEGIN BULK

\$  
PARAM,COUPMASS,1  
PARAM,AUTOSPC,YES  
PARAM,GRDPNT,0  
PARAM,USETPRT,0  
PARAM,NEWSEQ,-1

\$  
EIGR,10,MGIV,,1500.  
\$ASET,123,1,THRU,20000  
\$SPC1,11,123456,530

GRID,1,,0.00,0.,0.0,0  
GRID,2,,1.04,0.,0.5,0  
GRID,3,,5.46,0.,0.5,0  
GRID,4,,10.46,0.,0.5,0  
GRID,5,,15.00,0.,0.5,0  
GRID,6,,19.88,0.,0.5,0  
GRID,7,,20.6133,0.,0.5,0  
GRID,8,,23.17,0.,0.5,0

\$  
CBEAM,1,1,1,2,1.,1.,0.  
CBEAM,2,2,2,3,1.,1.,0.  
CBEAM,3,2,3,4,1.,1.,0.  
CBEAM,4,2,4,5,1.,1.,0.  
CBEAM,5,2,5,6,1.,1.,0.  
CBEAM,6,3,6,7,1.,1.,0.  
CBEAM,7,3,7,8,1.,1.,0.

\$  
PBEAM,1,101,0.429,,.169E-2,,.237E-2,,.365E-2  
PBEAM,2,102,,.141E-1,,.474E-2,,.474E-2,,.422E-2  
PBEAM,3,103,,.151,,.849E-3,,.998E-3,,.134E-2

\$  
MAT1,101,,.144E10,.56E9,,5.26  
MAT1,102,,.144E10,.56E9,,2.9  
MAT1,103,,.144E10,.56E9,,5.26

\$  
ENDDATA

---

A sketch of link four including the externally applied loads at the joints can be seen in Figure 3.2. Here, the strain gage is located at node seven. There are six degrees of freedom per node with a total of forty-eight degrees of freedom for the entire link. If substructure one is considered, the link would be cut at the strain gage location (node 7), resulting in a substructure containing six elements and a total of forty-two degrees of freedom. This substructure will be used to determine the loads at the strain gage location by using the following value of  $x_1$  :

$$x_1 = L x \quad (3.5)$$

where

$$L = [1 \mid 0] \quad (3.6)$$

For Eqs. (3.5) and (3.6),

$x_1$  = First Forty-Two Degrees of Freedom of  $x$  (42 x 1)

$1$  = Identity Matrix (42 x 42)

$0$  = Zero Matrix (42 x 6)

$x$  = Total Number of Degrees of Freedom of the Link (48 x 1)

$L$  = Locator Matrix (42 x 48)

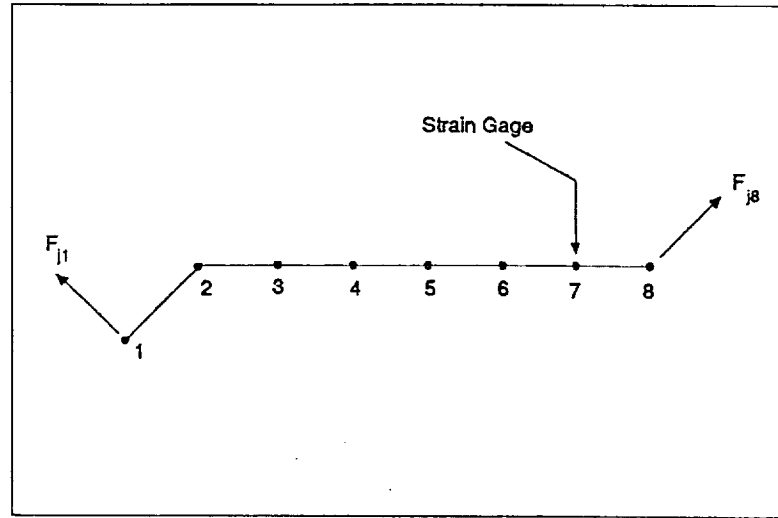


Figure 3.2: Seven Elements of Link Four

By substituting Eq. (3.5) into Eq. (3.3), the following equation can be obtained for the interface forces on substructure one of link four:

$$F_i = M_i L \ddot{x} + C_i L \dot{x} + K_i L x \quad (3.7)$$

Three variables used in the RMS code representing the mass, damping, and stiffness matrices in Eq. (3.7) are defined as the following:

$$ALOAD = M_i L \quad (3.8)$$

$$CLOAD = C_i L \quad (3.9)$$

$$KLOAD = K_i L \quad (3.10)$$

Therefore, the equation for the interface forces incorporating RMS code variables becomes

$$F_i = (ALOAD) \ddot{x} + (CLOAD) \dot{x} + (KLOAD) x \quad (3.11)$$

### 3.1.2 Algorithm / Software Description

The internal force/moment equations for links three and four are incorporated into two subroutines within the RMS code. This section will present a brief description of the subroutines and a definition of the variables used in the code.

Subroutine LDASPIN reads in the necessary input for the RMS dynamics. Within this routine, the fixed number of bodies and rigid degrees of freedom are prescribed, and the user is also allowed to define the flexible body information. The variable of interest in the strain gage force calculations, NUMFLXDOF, is read in from an input file called *flxbod.dat*. NUMFLXDOF is the number of component modes of flexible body degrees of freedom. This value will be used in the subroutine SGLOAD\_B to calculate  $\mathbf{x}$ ,  $\dot{\mathbf{x}}$ , and  $\ddot{\mathbf{x}}$  from Eq. (3.7).

Subroutine SGLOAD\_B computes the loads at the strain gage locations for links three and four. An input file is used to read in A\_LOAD, C\_LOAD, and K\_LOAD for link three (body 7) and link four (body 9). These values coincide with the mass, damping, and stiffness matrices, respectively. The code computing the components of the strain gage force vector in Eq. (3.11) is shown below:

$$\begin{aligned} F\_SHOULDER\_B7(I) &= F\_SHOULDER\_B7(I) + A\_LOAD\_3(I,J) \\ &\quad *X\_ACCEL\_3(J) + C\_LOAD\_3(I,J)*X\_VEL\_3(J) + \\ &\quad K\_LOAD\_3(I,J)*X\_DISPL\_3(J) \\ \\ T\_SHOULDER\_B7(I) &= T\_SHOULDER\_B7 + A\_LOAD\_3(I+3,J) \\ &\quad *X\_ACCEL\_3(J) + C\_LOAD\_3(I,J)*X\_VEL\_3(J) + \\ &\quad K\_LOAD\_3(I+3,J)*X\_DISPL\_3(J) \\ \\ F\_WRIST\_B9(I) &= F\_WRIST\_B9(I) + A\_LOAD\_4(I,J) \\ &\quad *X\_ACCEL\_4(J) + C\_LOAD\_4(I,J)*X\_VEL\_4(J) + \\ &\quad K\_LOAD\_4(I,J)*X\_DISPL\_4(J) \\ \\ T\_WRIST\_B9(I) &= T\_WRIST\_B9 + A\_LOAD\_4(I+3,J) \\ &\quad *X\_ACCEL\_4(J) + C\_LOAD\_4(I+3,J)*X\_VEL\_4(J) + \\ &\quad K\_LOAD\_4(I+3,J)*X\_DISPL\_4(J) \end{aligned}$$

where

F\_SHOULDER\_B7 - Strain gage force of link 3 (body 7) defined in 24CROSS.INC  
 T\_SHOULDER\_B7 - Strain gage torque of link 3 (body 7) defined in  
 24CROSS.INC  
 F\_WRIST\_B9 - Strain gage force of link 4 (body 9) defined in 24CROSS.INC  
 T\_WRIST\_B9 - Strain gage torque of link 4 (body 9) defined in 24CROSS.INC  
 A\_LOAD\_3 - Local real\*8 array of mass matrix of link 3  
 C\_LOAD\_3 - Local real\*8 array of damping matrix of link 3  
 K\_LOAD\_3 - Local real\*8 array of stiffness matrix of link 3  
 A\_LOAD\_4 - Local real\*8 array of mass matrix of link 4  
 C\_LOAD\_4 - Local real\*8 array of damping matrix of link 4  
 K\_LOAD\_4 - Local real\*8 array of stiffness matrix of link 4  
 X\_ACCEL\_3 - Local real\*8 array of acceleration vector of link 3  
 X\_VEL\_3 - Local real\*8 array of velocity vector of link 3  
 X\_DISPL\_3 - Local real\*8 array of displacement vector of link 3  
 X\_ACCEL\_4 - Local real\*8 array of acceleration vector of link 4  
 X\_VEL\_4 - Local real\*8 array of velocity vector of link 4  
 X\_DISPL\_4 - Local real\*8 array of displacement vector of Link 4

### 3.1.3 Constraint Modes

In CMS techniques, discretized substructures are partitioned into boundary (b) and interior (I) physical degrees of freedom as shown in Figure 3.3. The boundary degrees of freedom represent interface or common degrees of freedom with adjoining substructures. For an unrestrained structure with rigid body motion, the boundary degrees of freedom can be further divided into rigid body (r) and excess (e) coordinates.



The linear equations of motion for the substructure can be partitioned as

$$\begin{bmatrix} \mathbf{M}_{ii} & \mathbf{M}_{ib} \\ \mathbf{M}_{bi} & \mathbf{M}_{bb} \end{bmatrix} \begin{pmatrix} \ddot{\mathbf{x}}_i \\ \ddot{\mathbf{x}}_b \end{pmatrix} + \begin{bmatrix} \mathbf{K}_{ii} & \mathbf{K}_{ib} \\ \mathbf{K}_{bi} & \mathbf{K}_{bb} \end{bmatrix} \begin{pmatrix} \mathbf{x}_i \\ \mathbf{x}_b \end{pmatrix} = \begin{pmatrix} \mathbf{F}_i \\ \mathbf{F}_b \end{pmatrix} \quad (3.12)$$

As defined by Craig, a constraint mode is the resulting static deformation of a structure with a unit displacement on one of the boundary degrees of freedom while the other boundary coordinates are held fixed and the interior coordinates are free. This is described by Eq. (3.12) where  $\mathbf{R}_{bb}$  is the set of reaction forces at the boundary degrees of freedom and  $\mathbf{1}$  is a  $b \times b$  identity matrix.

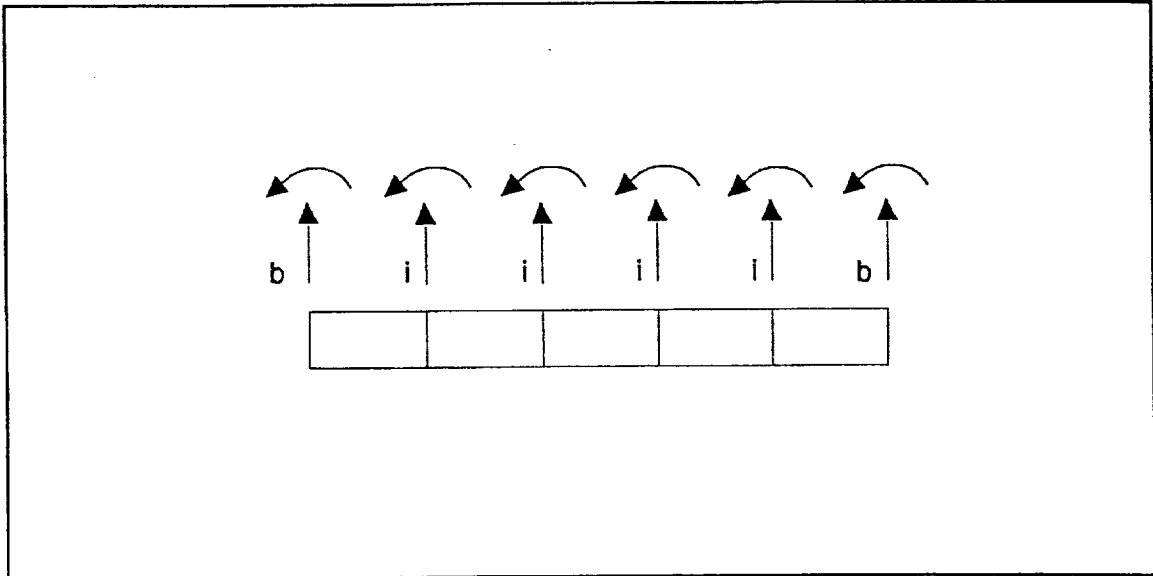


Figure 3.3: Discretized Substructure

$$\begin{bmatrix} \mathbf{K}_{ii} & \mathbf{K}_{ib} \\ \mathbf{K}_{bi} & \mathbf{K}_{bb} \end{bmatrix} \begin{pmatrix} \Psi_{ic} \\ \mathbf{1} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{R}_{bb} \end{pmatrix} \quad (3.13)$$

The resulting shapes,  $\Psi_{ic}$ , can be found from the top row partition of Eq. (3.13).

$$\Psi_c = \begin{pmatrix} \Psi_{ic} \\ \Psi_{bc} \end{pmatrix} + \begin{pmatrix} -K_{ij}^{-1} K_{ib} \\ \mathbf{1} \end{pmatrix} \quad (3.14)$$

where  $K_{ij}$  and  $K_{ib}$  are from the partitioned substructure stiffness matrix.

Constraint modes are typically augmented with fixed interface normal modes. These mode shapes are computed with the partitioned mass and stiffness matrices, holding the boundary degrees of freedom clamped, from the following eigenvalue problem:

$$(K_{ij} - \omega^2 M_{ij}) \phi_{in} = 0 \quad (3.15)$$

where  $\omega^2$  is a diagonal matrix whose elements are the square of the fixed interface natural frequencies, and  $\phi_{in}$  is the  $i \times i$  matrix of normal mode shapes. These modes are usually normalized such that

$$\phi_{in}^T M_{ij} \phi_{in} = \mathbf{1} \quad (3.16)$$

where  $\mathbf{1}$  is an  $i \times i$  identity matrix.

The substructure degrees of freedom are written in terms of the constraint and fixed interface modes as

$$\begin{pmatrix} X_i \\ X_b \end{pmatrix} = \begin{bmatrix} \Psi_{ic} & \phi_{in} \\ \mathbf{1} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \eta_c \\ \eta_n \end{pmatrix} \quad (3.17)$$

Note that the constraint modes are naturally linearly independent with respect to fixed interface normal modes.

All of the component modes used in RMS simulation were orthogonalized with respect to the six rigid body modes of the free-free substructure and each other using a modified form of the Gram-Schmidt technique. Orthogonalizing the component modes with

respect to the substructure rigid body modes serves a number of purposes. First, the mass integral terms  $\Gamma_0$  and  $\Gamma_1$  of LDASP are zero for this condition. These terms may be eliminated from the equations of motion and decrease the simulation execution time.

Second, the flexible body deformations computed by the assumed modes expression represent true bending displacements. If the assumed modes contain rigid body components, then so will the deformation terms. This will also alter the values of the system rigid body degrees of freedom through coupling from the non-zero mass integrals. The resulting solutions are not wrong; however, the system variables no longer truly describe rigid body and flexible body motion.

Third, this procedure is necessary to eliminate assumed modes, or components of assumed modes, which may span rigid body motion already accounted for in the system rigid body degrees of freedom, such as RMS joint and base vehicle motion. This problem will manifest itself as a singularity in the solution of the system equations of motion since the rigid body coordinates and modal degrees of freedom are describing the same motion. A set of constraint modes for a given substructure often contains rigid body modes or are spanned, in part, by a set of rigid body modes.

To describe the modified Gram-Schmidt procedure, the candidate component modes  $\phi_n$  for use in the assumed modes technique are augmented with the six substructure rigid body modes  $\phi_r$ .

$$\mathbf{u} = [\phi_r : \phi_n] \quad (3.18)$$

The vectors of  $\mathbf{u}$  are normalized with respect to the substructure mass matrix  $\mathbf{M}$  to have a generalized mass of one.

$$\mathbf{v}_j = \frac{\mathbf{u}_j}{\sqrt{\mathbf{u}_j^T \mathbf{M} \mathbf{u}_j}} \quad (3.19)$$

The orthogonalized vectors are labeled as  $\Psi$ . A sweeping matrix to mass orthogonalize the  $j^{\text{th}}$  vector of  $\mathbf{v}$  respect to the previous  $j-1$  vectors is

$$S_{j-1} = 1 - \frac{\psi_1 \psi_1^T M}{\psi_1^T M \psi_1} - \frac{\psi_2 \psi_2^T M}{\psi_2^T M \psi_2} - \dots - \frac{\psi_{j-1} \psi_{j-1}^T M}{\psi_{j-1}^T M \psi_{j-1}} \quad (3.20)$$

The trial vector  $\bar{\psi}_j$  is computed from  $S_{j-1}$  as

$$\bar{\psi}_j = S_{j-1} v_j \quad (3.21)$$

If the generalized mass of  $\bar{\psi}_j$  is less than a user defined tolerance, typically  $10^{-6}$ , this vector is deleted from the set of component modes. Otherwise, it is normalized to have a generalized mass of one.

$$\psi_j = \frac{\bar{\psi}_j}{\sqrt{\bar{\psi}_j^T M \bar{\psi}_j}} \quad (3.22)$$

The tolerance is essentially a numerical definition for linear independence of the vectors. Without this step, the normal Gram-Schmidt algorithm can lead to vectors which are not linearly independent for all practical purposes.

For links three and four of the RMS, eighteen degrees of freedom were selected as boundary coordinates and the remaining thirty were designated as interior coordinates. The boundary degrees of freedom for each member consist of the six coordinates at the end nodes and the six at the strain gage node. This results in eighteen constraint modes and thirty fixed interface normal modes for each member. The orthogonalization process eliminates six constraint modes. For the RMS simulation, twelve constraint modes and four fixed interface normal modes were used as component modes for links three and four (bodies seven and nine in LDASP). The flex body data for LDASP is stored in the ASCII file FLXBOD.DAT. The number of component modes used in the simulation may be reduced by editing this file. Two other versions of this file reside on the Alliant computer system ALLI in the directory "/usr/sim/exec/rms/sg/constr" named FLXBOD4.4 and

FLXBOD12.12. These files have already been changed to use four and twelve component modes for the RMS members.

### 3.1.4 Verification Cases

In order to verify the algorithm and flex body input data developed to compute internal loads, a new static mode of operation labeled validation case #30 was created. The LDASP model cantilevers the RMS to ground through a translational and torsional spring at the payload. The joints are held in place with the locking springs. A unit force or moment is applied to the center of mass of the base vehicle along/about the axis specified by the user. The load is initially zero and ramps to a value of one over ten seconds.

For the first set of tests, the original model with free-free component modes was run varying the number of system modes used in the solution. The runs were then repeated using the modified mode acceleration method and varying FREQLIM to change the number of system modes below the low pass filter cut off frequency. These initial tests of the mode acceleration method also used the free-free component modes. Figure 3.4 depicts the internal force along x in the shoulder member due to an external unit load applied in x.

In the original model using five system modes, the computed load was less than .2 pounds. With ten system modes, the response grew to almost 1.1 pounds. However, with the new model, the correct answer was found with only five system modes. Additional system modes had no impact on the internal load. These results demonstrate the advantages of the modified mode acceleration method over the mode displacement technique. Since fewer system modes are required to converge to the correct response, the resulting simulation cycle time will be shorter. Figures 3.5 through 3.15 display the responses for loads applied in the other directions. The mode acceleration technique was as good, if not better, than the mode displacement approach for all cases. Note that the external load, although slowly applied, excites modes in the RMS for some cases. This can be seen in Figures 3.5 and 3.6. The resulting steady state value, after the vibrations dampen out, is still unity.

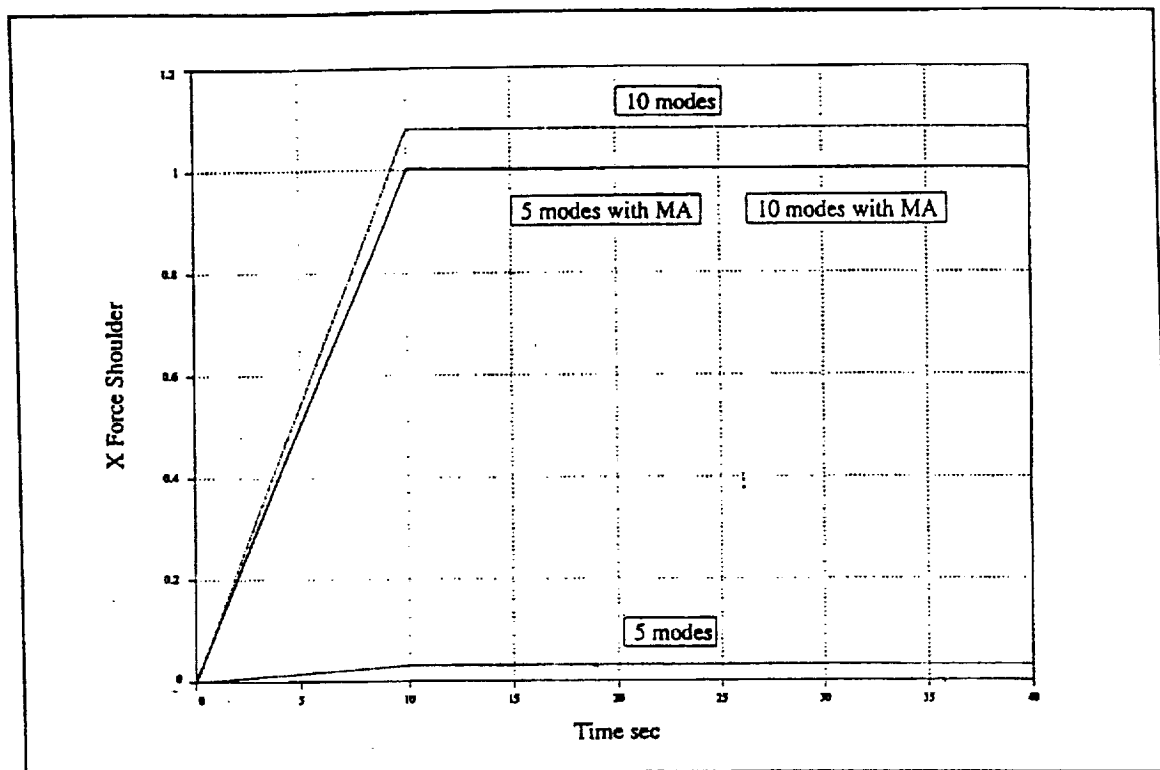


Figure 3.4: Shoulder X Force Static Validation Runs

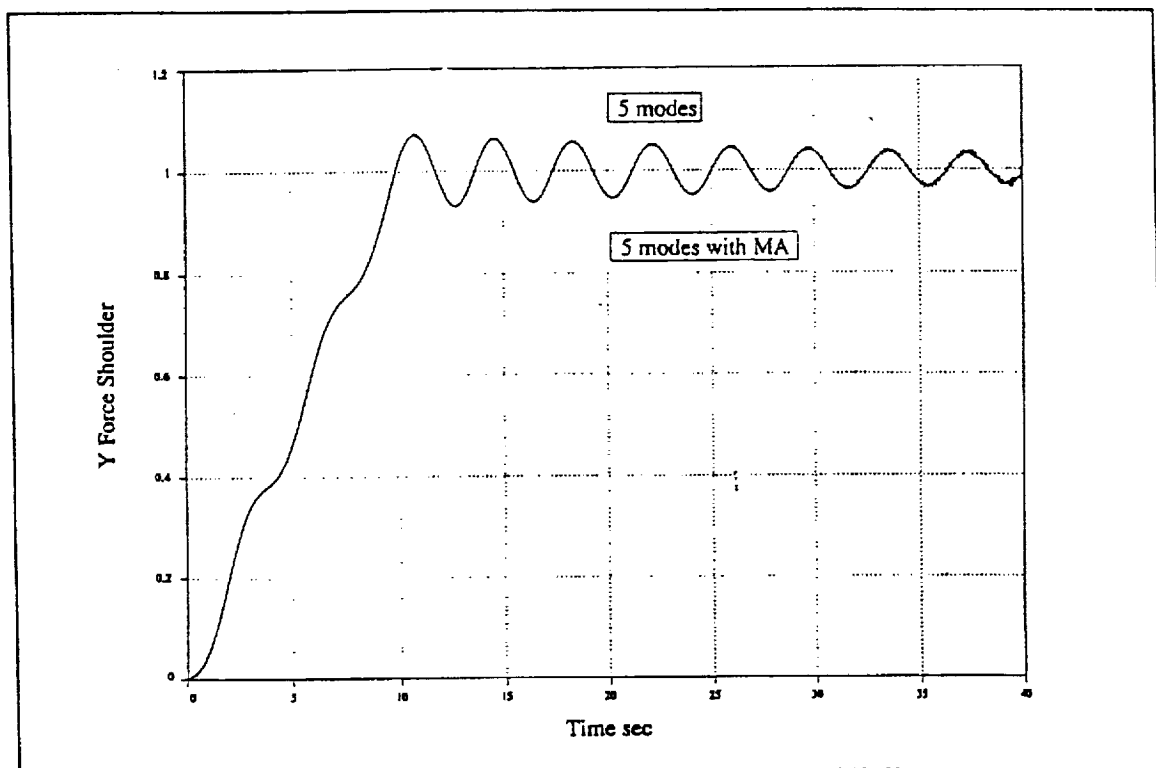


Figure 3.5: Shoulder Y Force Static Validation Runs

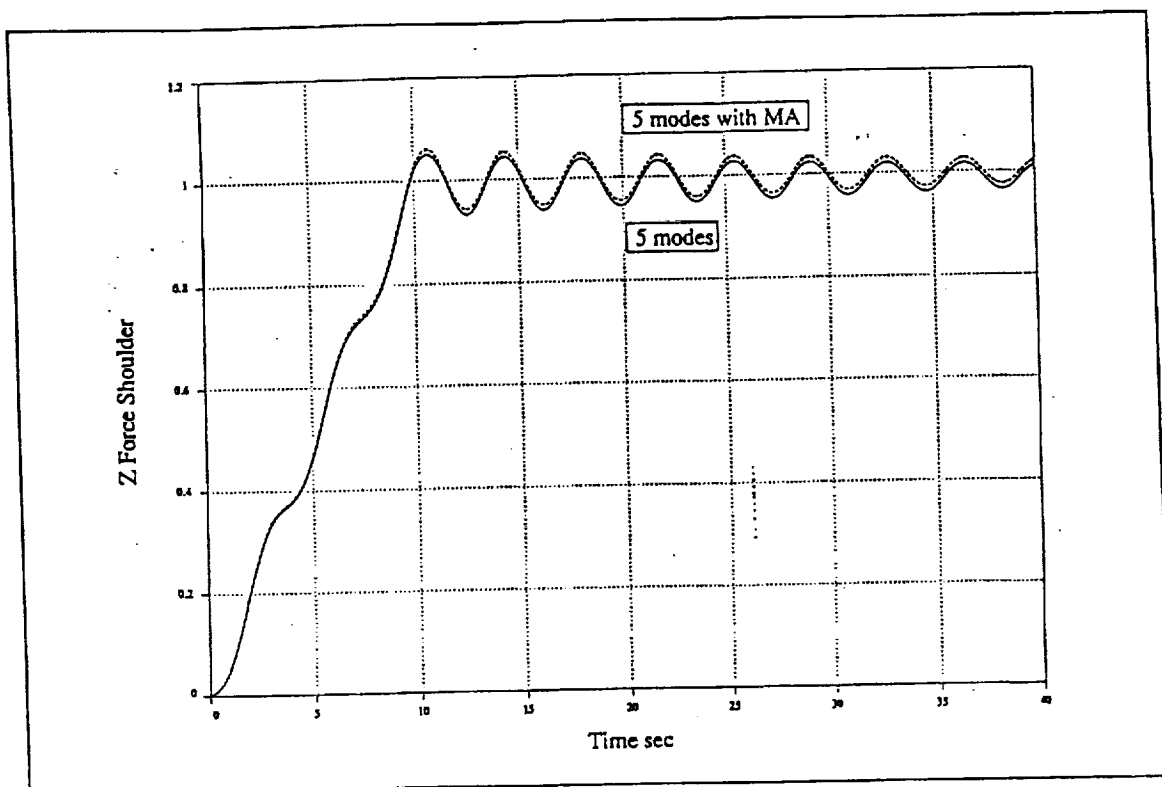


Figure 3.6: Shoulder Z Force Static Validation Runs

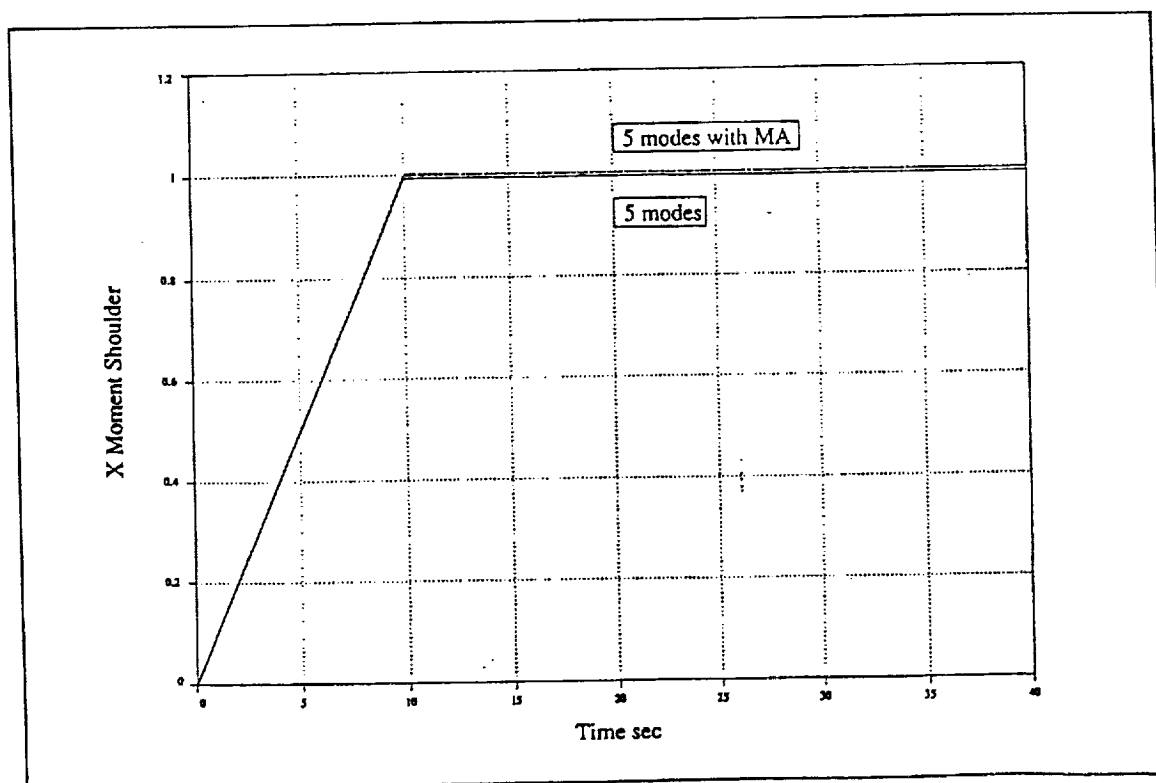


Figure 3.7: Shoulder X Moment Static Validation Runs

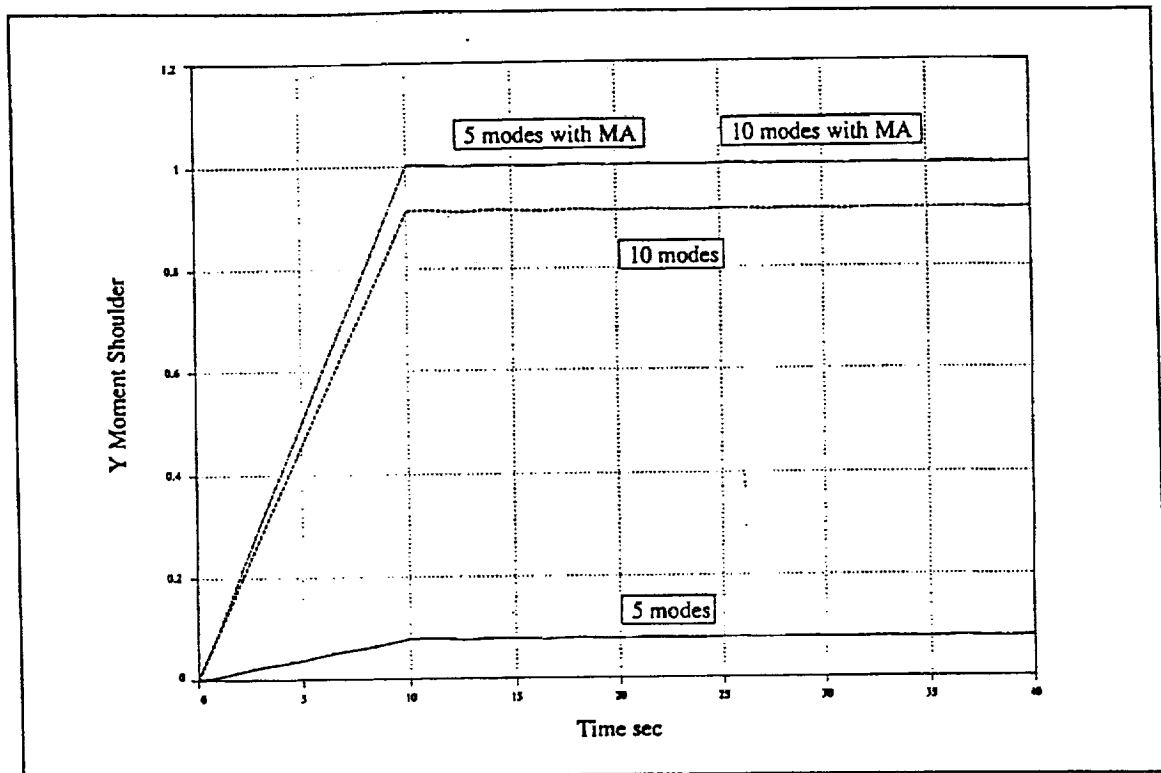


Figure 3.8: Shoulder Y Moment Static Validation Runs

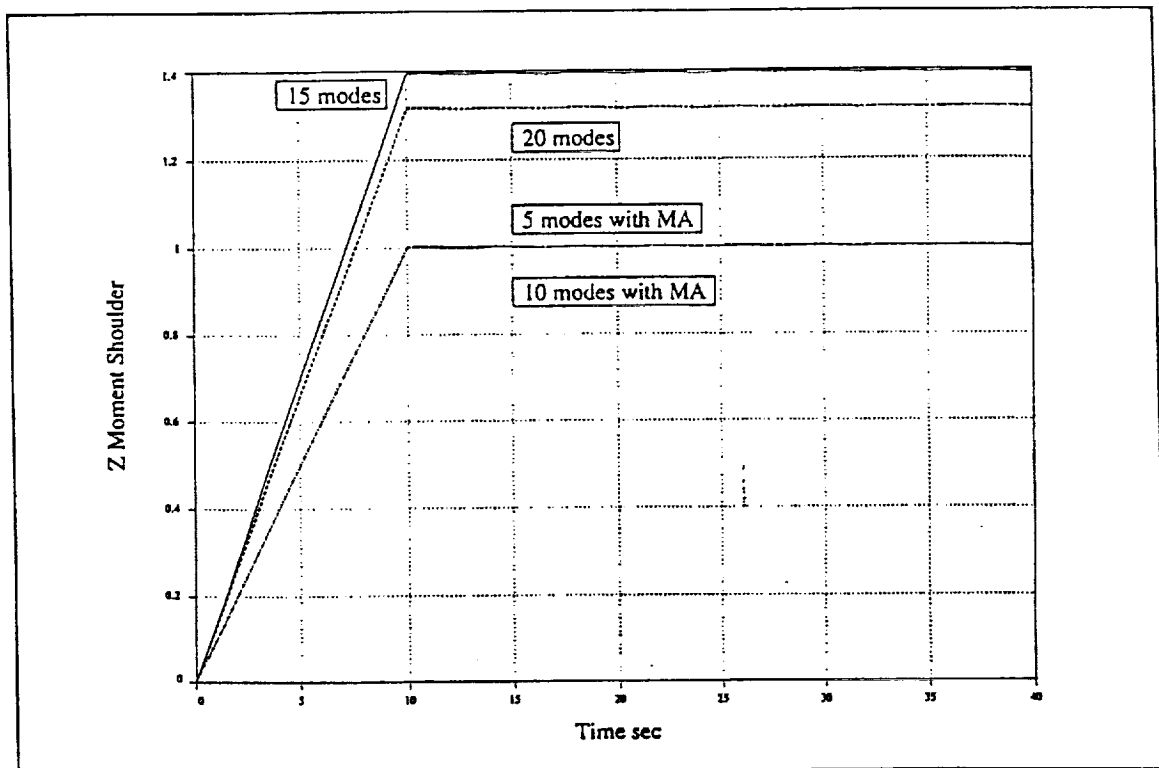


Figure 3.9: Shoulder Z Moment Static Validation Runs



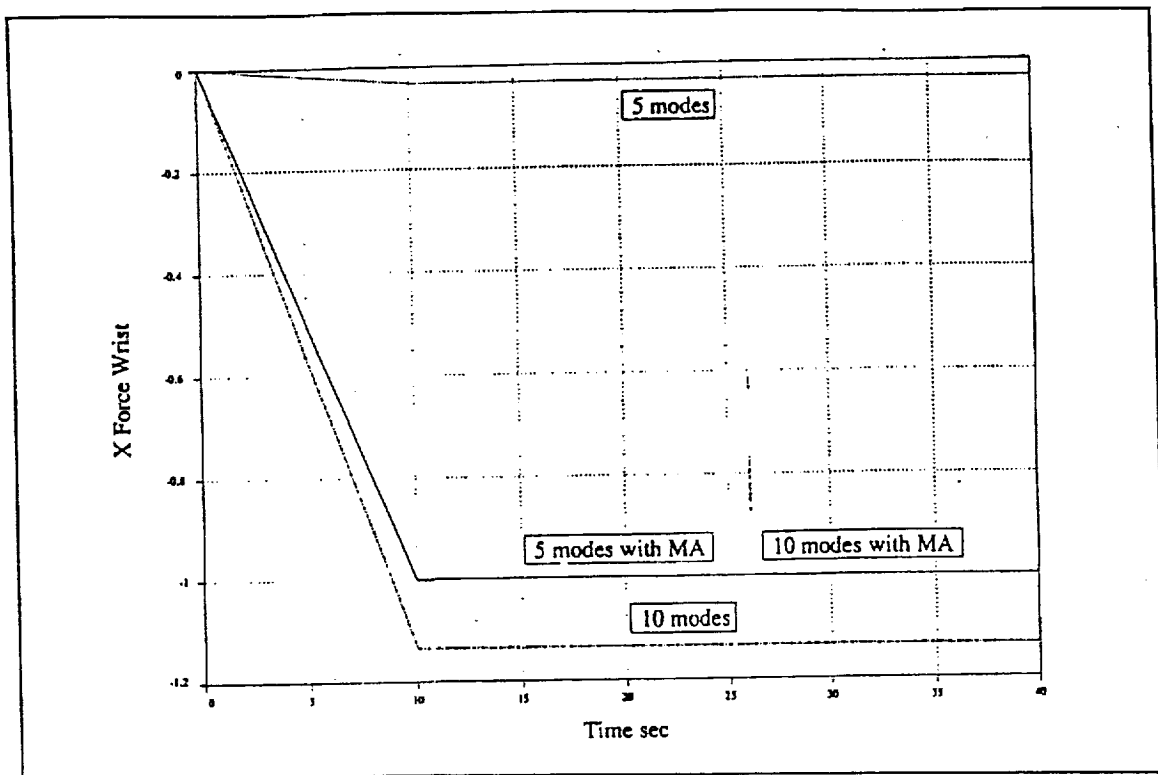


Figure 3.10: Wrist X Force Static Validation Runs

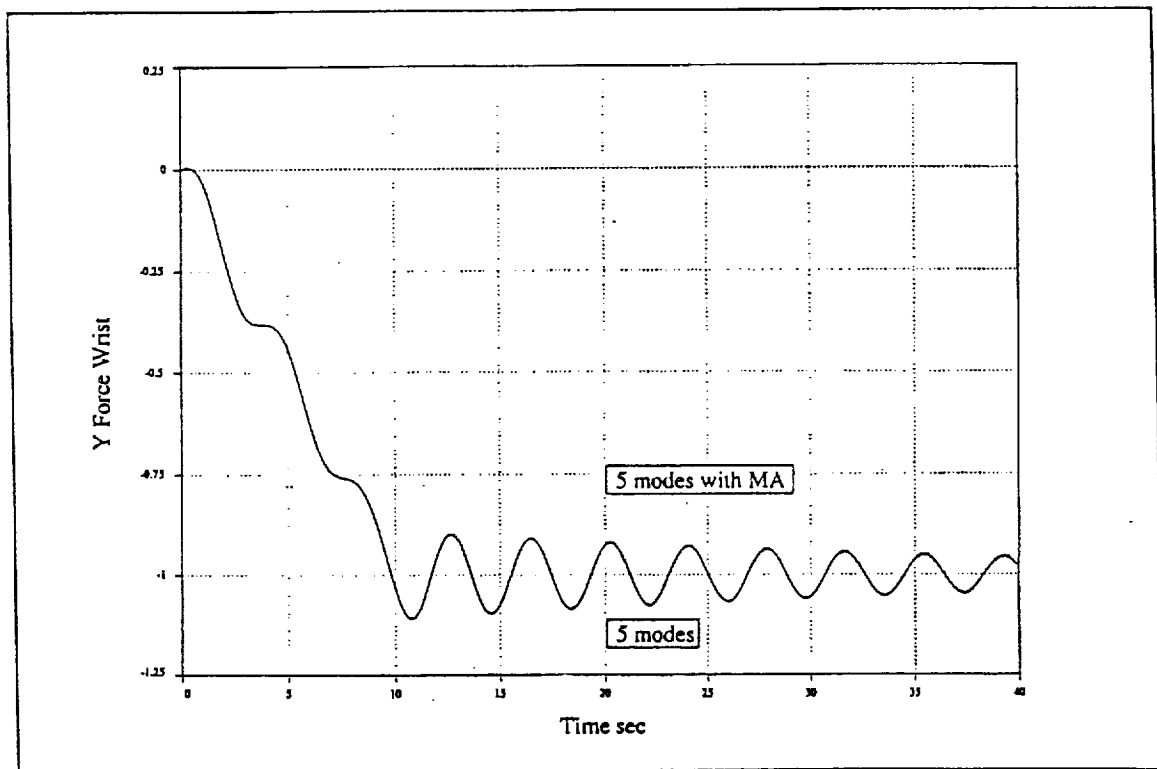


Figure 3.11: Wrist Y Force Static Validation Runs

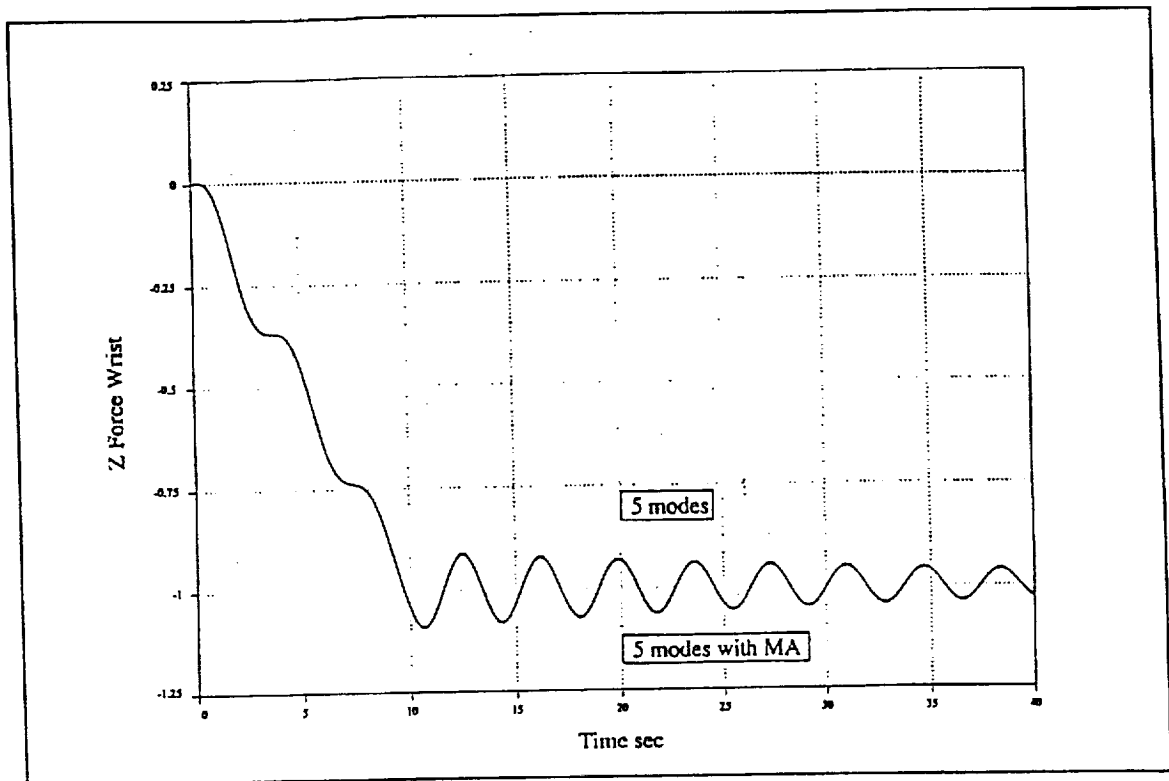


Figure 3.12: Wrist Z Force Static Validation Runs

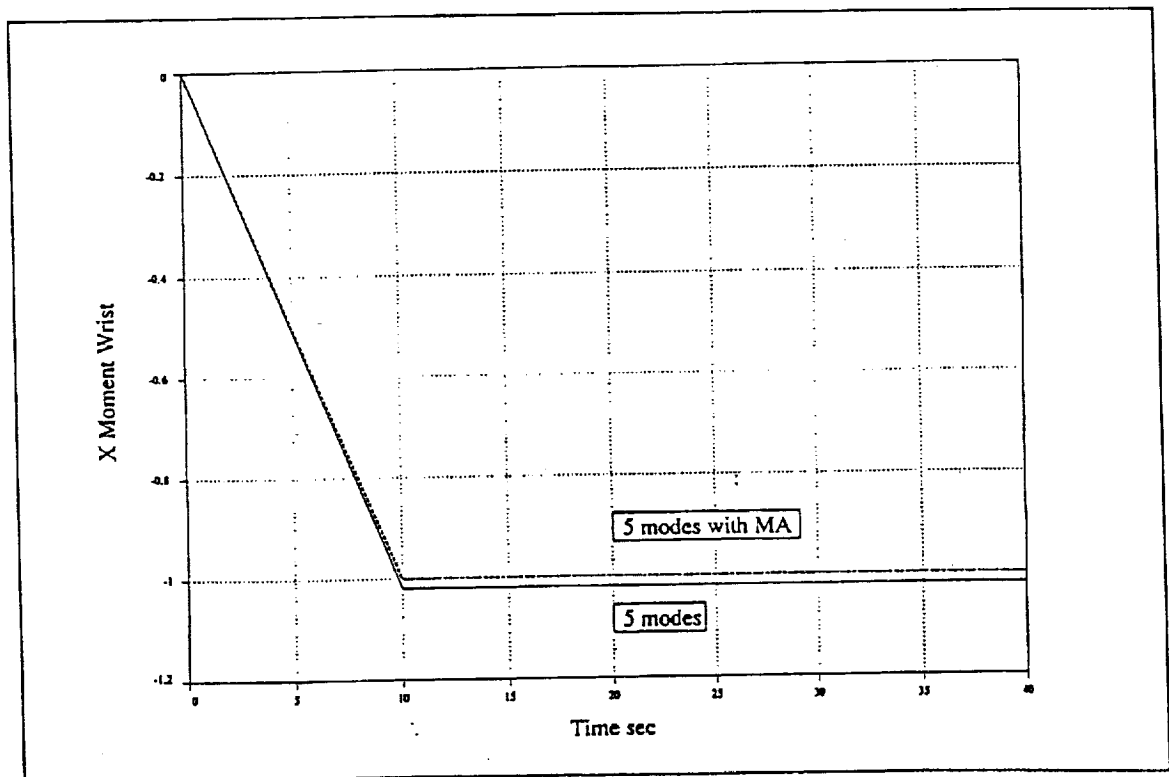


Figure 3.13: Wrist X Moment Static Validation Runs

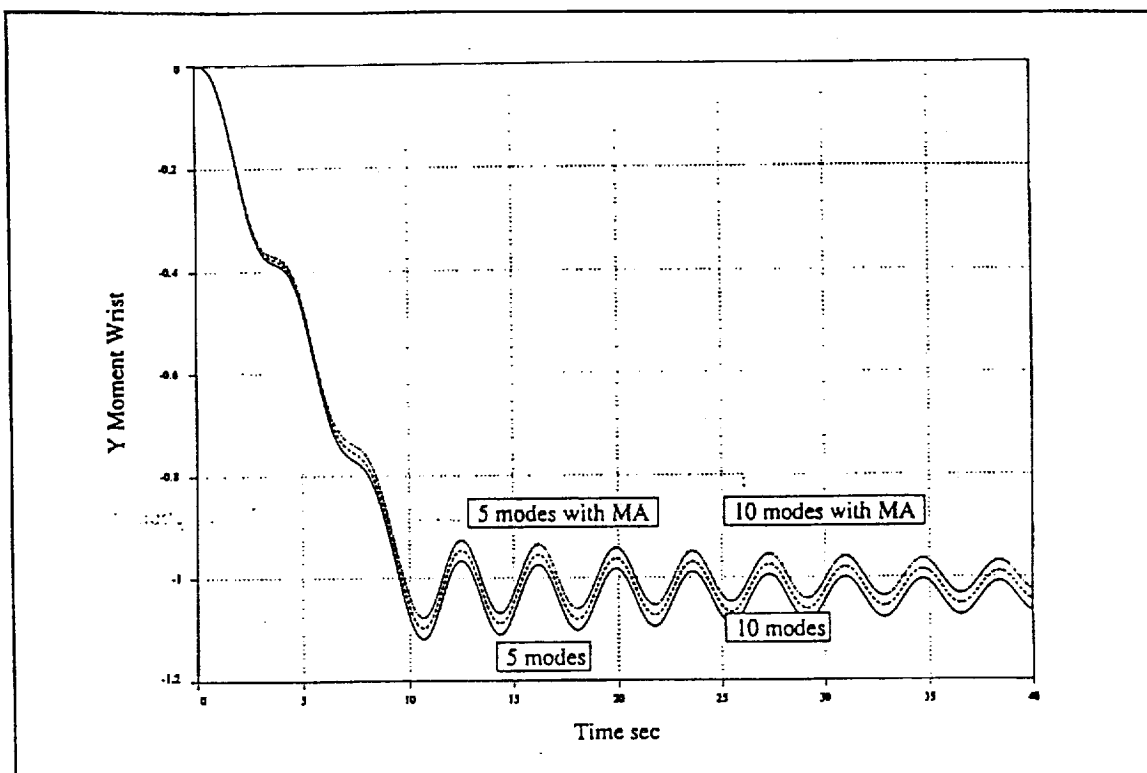


Figure 3.14: Wrist Y Moment Static Validation Runs

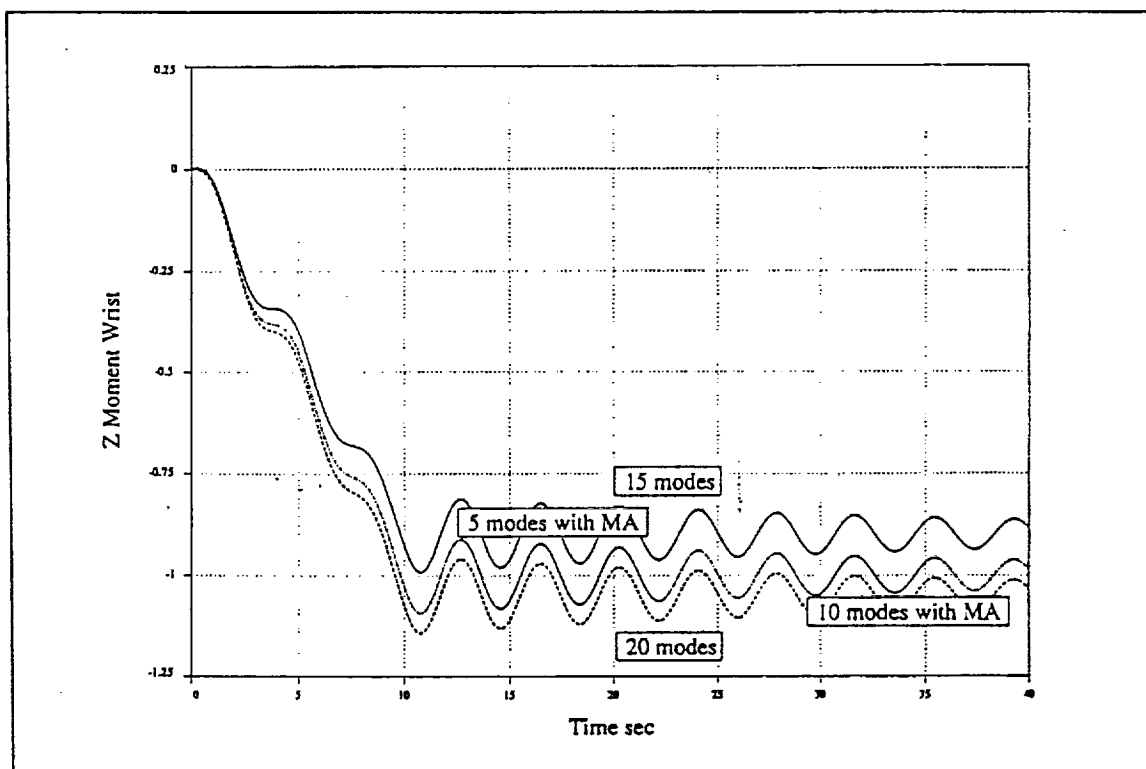


Figure 3.15: Wrist Z Moment Static Validation Runs

### 3.2 RMS Operation Termination Conditions

In order to incorporate the RMS validation cases into the real time software, a series of case control variables were developed and set in the subroutine INTERACT.

VT\_CSTART - Global real\*8 command start time defined in CONSTANT. INC

VT\_CEND - Global real\*8 command end time defined in CONSTANT. INC

VT\_BRAKES - Global real\*8 brake engage on time defined in CONSTANT. INC

VT\_TERM - Global real\*8 simulation end time defined in CONSTANT. INC

For the first six thruster firing P cases, the RMS brakes are engaged for all joints for the duration of the run. The RMS control system is essentially inactive for these runs since the motor field torques are set to zero when the brakes are on. P cases twelve through fourteen terminate with the safing condition. Under safing, the controller outputs zero torque after VT\_CEND and the joint coasts until the velocity drops below .5 degrees/second. At this point, the joint is placed in position hold mode and is actively held in place. The remaining joints are in position hold for the duration of the run. For P cases fifteen and sixteen, all joints are in position hold for the entire run.

The flight software keys off the variable SINGLE\_MODE to distinguish between the termination conditions for the P cases as well as the direct drive and single joint runs. The brakes on and position hold P cases have a SINGLE\_MODE value of two while the safing cases have a value of three.

The direct drive validation cases have a value of SINGLE\_MODE of two. The brakes are on for the entire run for all non-driven joints once the drive command is initiated. The selected joint is driven at a constant rate through a command hard coded in SERVO. The brake for the driven joint is engaged at the end of the command time VT\_CEND.

The single joint validation cases use a value of SINGLE\_MODE of one. In the single joint mode, all non-driven joints are in position hold for the length of the drive command. The selected joint is driven at a constant rate through commands computed in the subroutine SCF.

Only minor changes were made to input data in the MITL software for validation cases. The joystick deadband variables `THC_NULL_DEADBAND` and `RHC_NULL_DEADBAND` were set to values of eight in the subroutine MITL. The maximum joystick deflection arrays `THC_MAX_DEFLECT` and `RHC_MAX_DEFLECT` were set to ninety-three for all axes.

A new output routine, `OUTPUT_SV`, was developed to mix flight data from non-driven joints with simulation data for the driven joints to compute POR data for single joint validation runs. The POR computations are performed in the subroutine `KDG_VAL`. `KDG_VAL` is a copy of `KDG` with logic to read in joint angles and rates from flight data files. After the new POR data is written out by `OUTPUT_SV`, `KDG` is called to reset joint data used by the flight controller in the simulation.

### 3.3 Validation Mode Example Runs: Parallel vs. Flight Results

This section summarizes the examples that were run for each type of validation case. The plots for each of these cases are located in the appendices and illustrate comparisons between the flight data and the Parallel simulation results. Appendix A contains plots of the results of a P20 thruster firing validation case (Validation #5) which is terminated with brakes. These plots contain the load data from the shoulder and wrist strain gages. Appendix B includes plots showing the results of the D7 shoulder pitch direct drive case (Validation #10) terminated with brakes. Appendix C contains plots of the S7 wrist yaw single joint case (Validation #19) which is also terminated with brakes. The plots in Appendix B and C include motor rate, joint angle, POR position, attitude, velocity, rate, and load data from shoulder and wrist strain gauges. Appendix D includes plots showing the results of a M5 man-in-the-loop case (Validation #29) terminated with position hold. These plots contain the motor rate for each joint and the load data from the shoulder and wrist strain gages.

#### **4.0 CONCLUSION/RECOMMENDATIONS**

This report has presented the upgrades to the RMS simulation in support of the formal validation process. The addition of the MMAM and the constraint / fixed interface component modes has eliminated the drift problem and improved the stability of the simulation. The joint and motor friction models have been tuned to improve the correlation between the simulation responses and the flight data. The servo electronics state space model has been upgraded to match the block diagram presented in the Payload Simulation Database document. All twenty-nine flight to sim validation cases were run and graded. The overall grade for the simulation is 85.4. All changes to the model have been incorporated into the real time simulation as well as ACDS.

Future enhancements to the RMS simulation include the deletion of the long integration cycle and the reduction of the number of component modes and resulting model size to decrease cycle time.

**APPENDIX A**  
**P20 THRUSTER FIRING VALIDATION CASE**

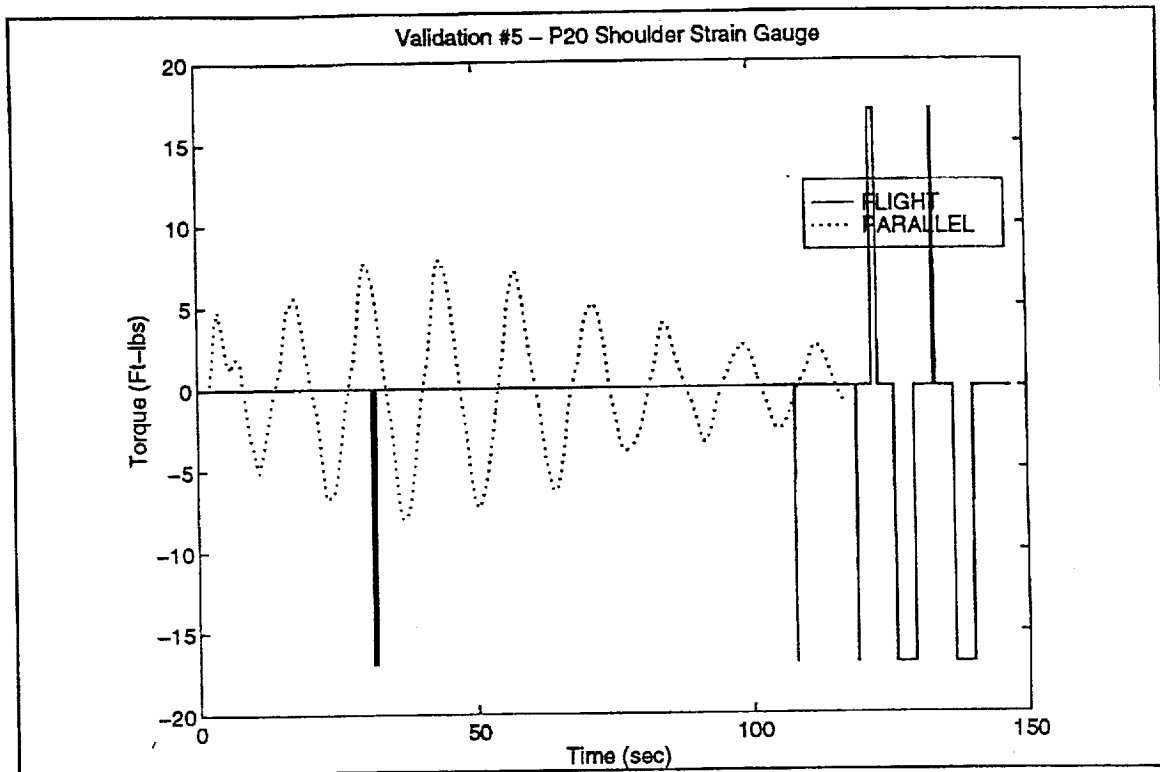


Figure A1: Validation #5 - P20 Shoulder Strain Gauge

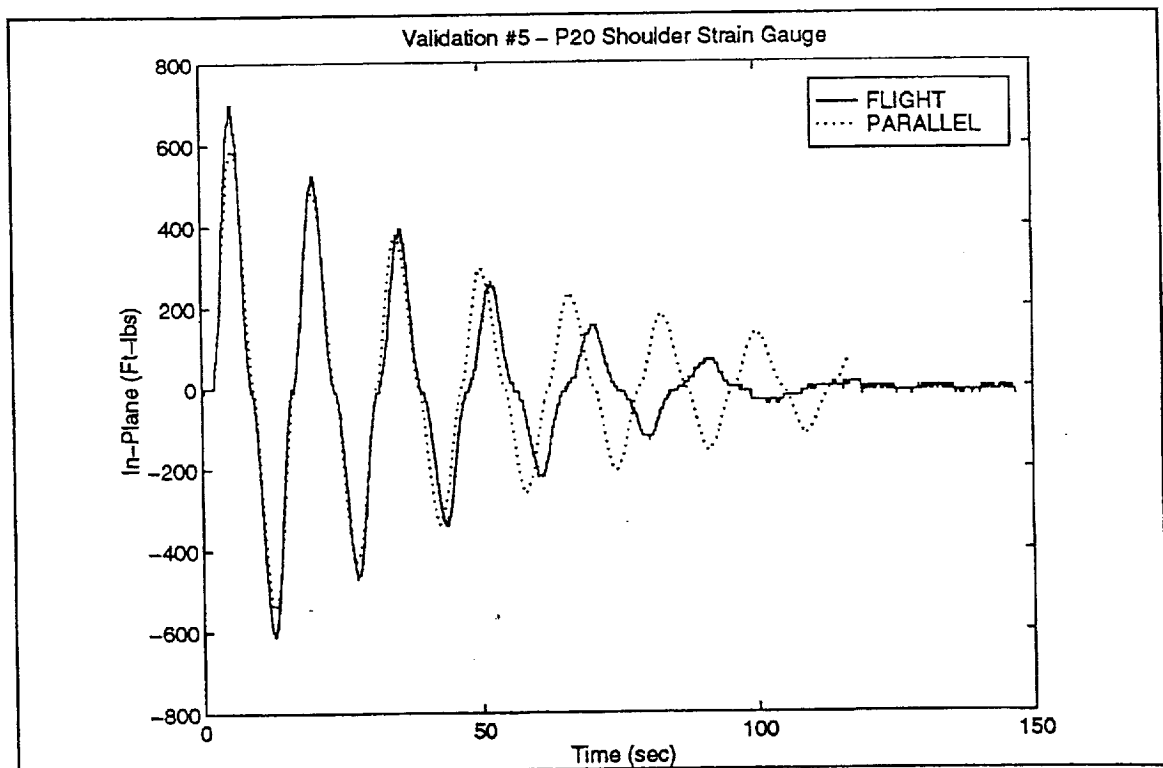


Figure A2: Validation #5 - P20 shoulder Strain Gauge



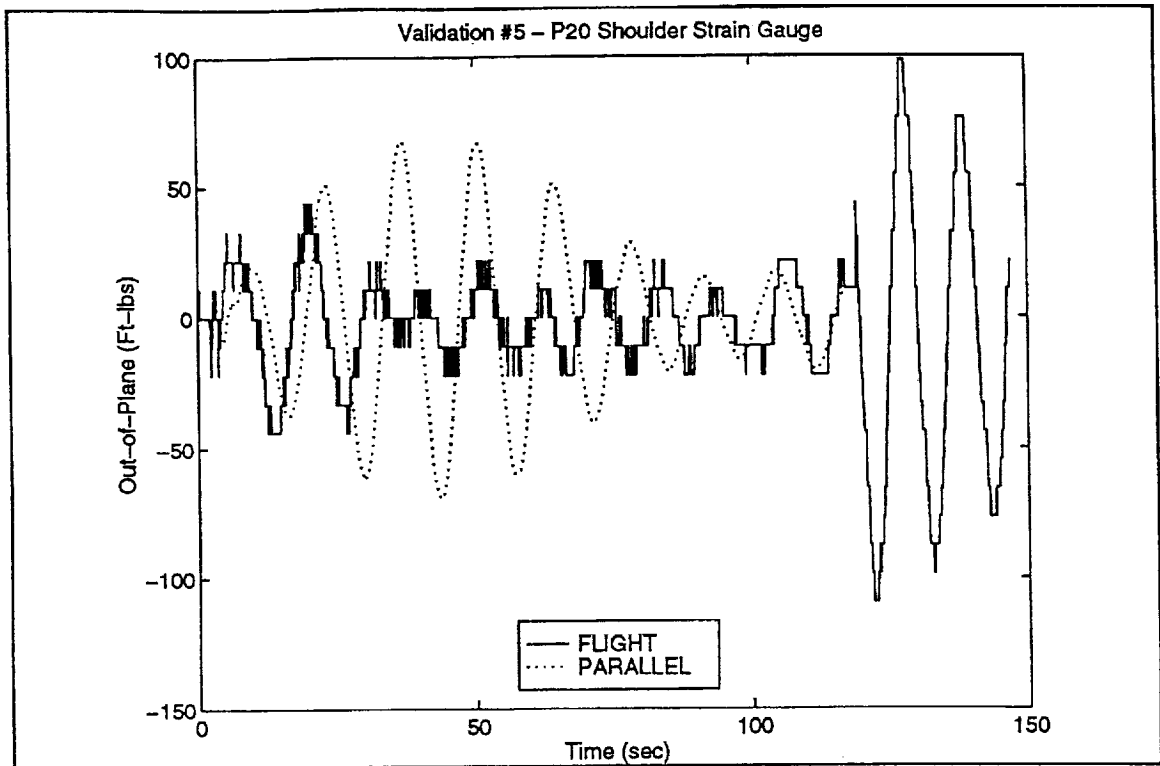


Figure A3: Validation #5 - P20 Shoulder Strain Gauge

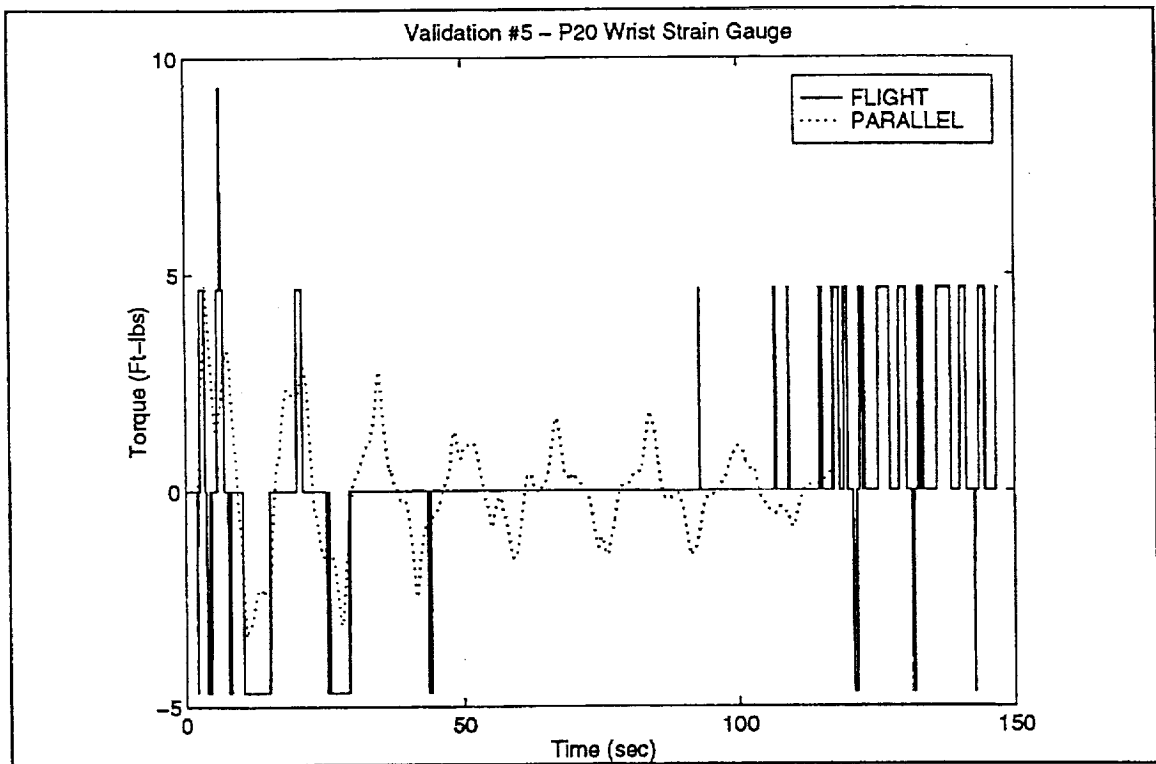


Figure A4: Validation #5 - P20 Wrist Strain Gauge

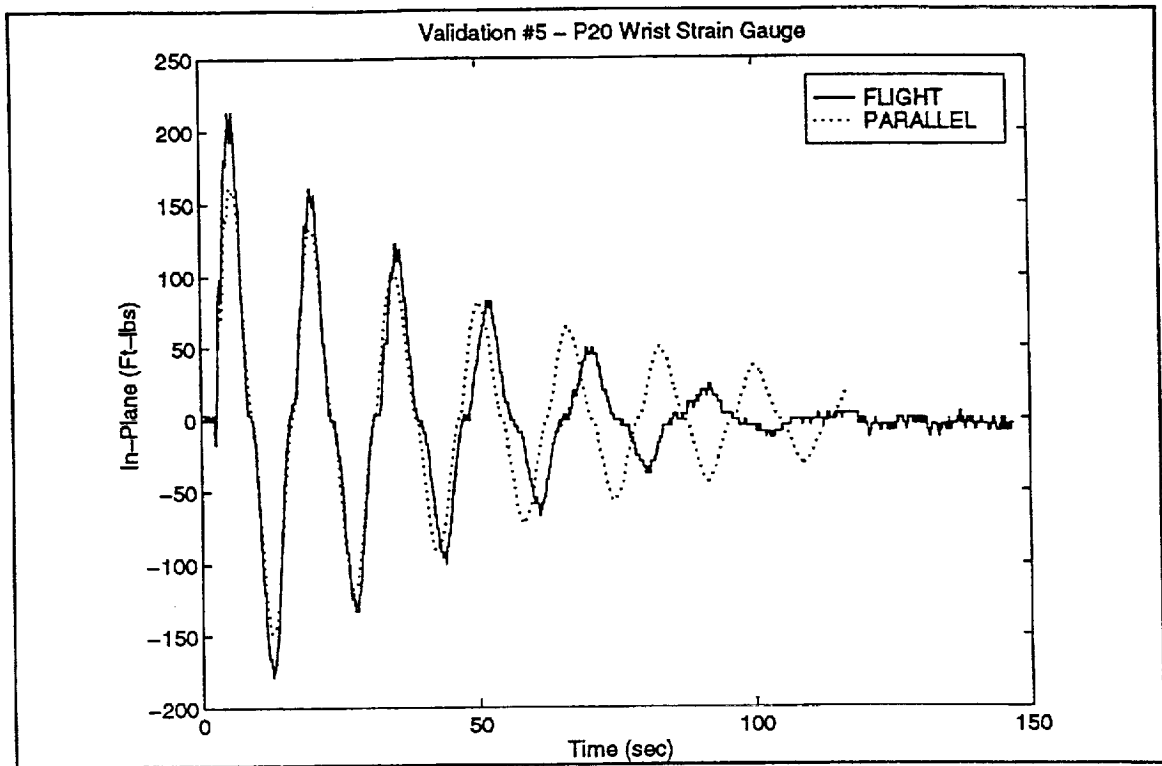


Figure A5: Validation #5 - P20 Wrist Strain Gauge

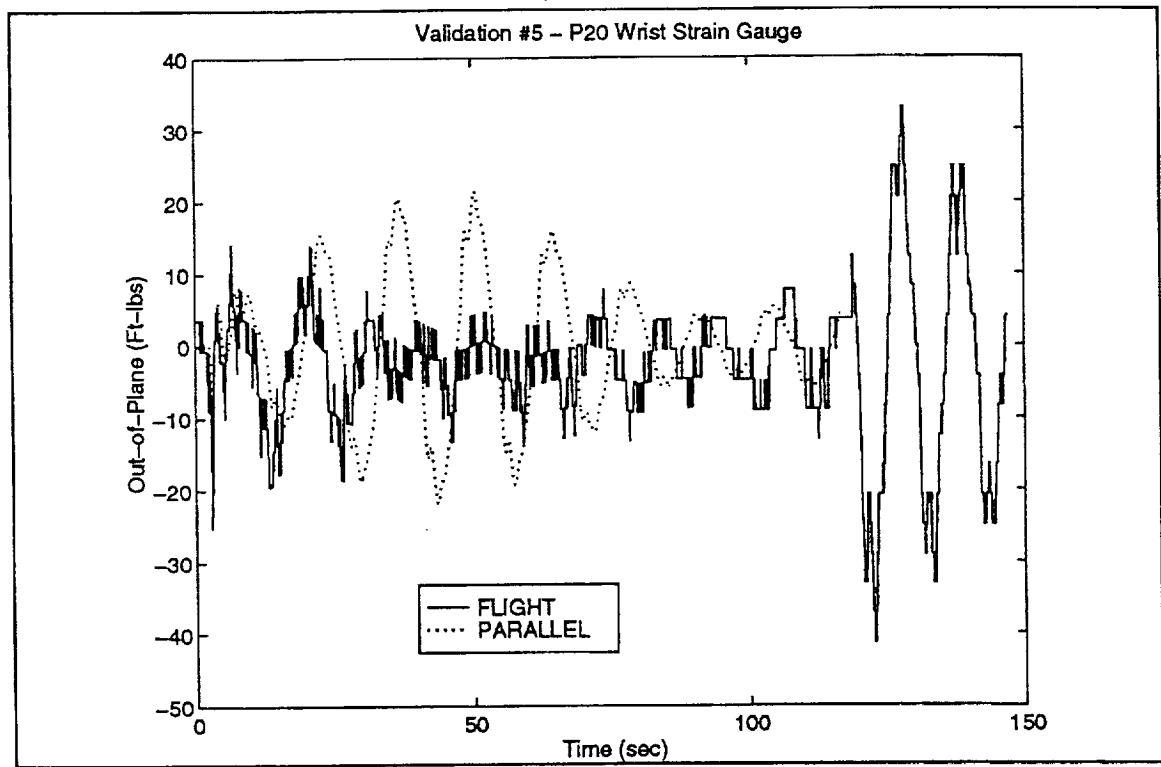


Figure A6: Validation #5 - P20 Wrist Strain Gauge

**APPENDIX B**  
**D7 DIRECT DRIVE VALIDATION CASE**

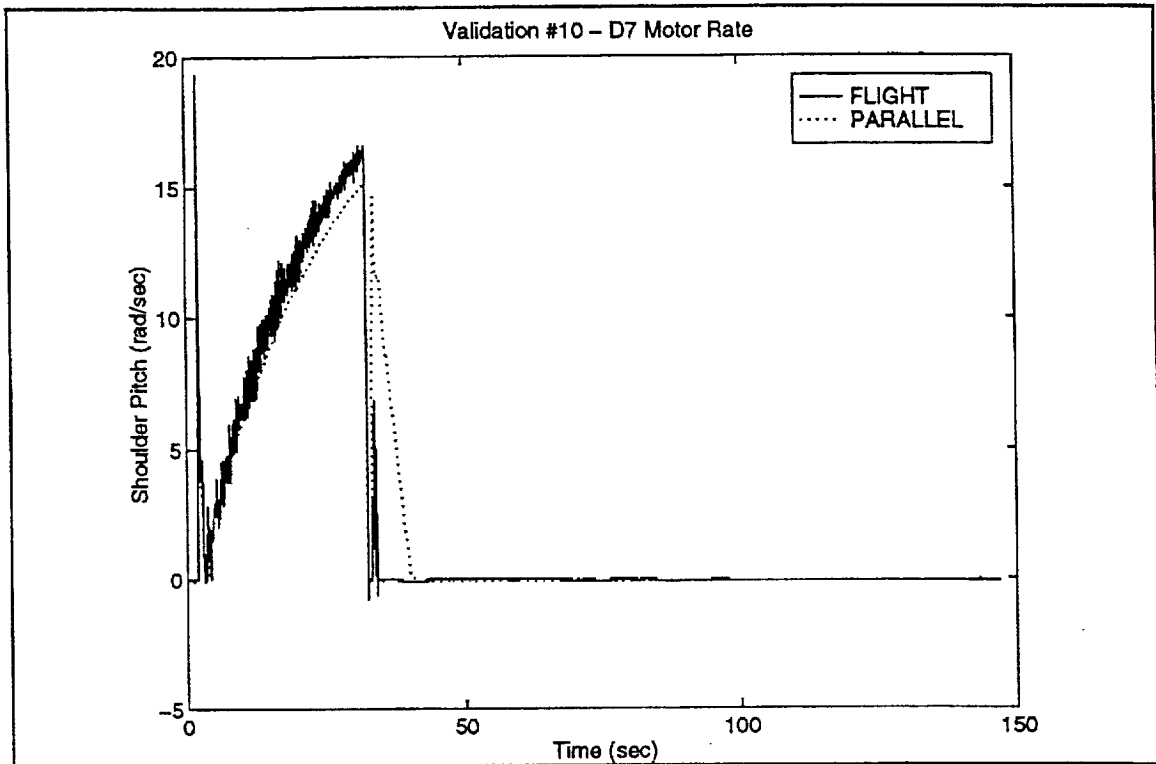


Figure B1: Validation #10 - D7 Motor Rate

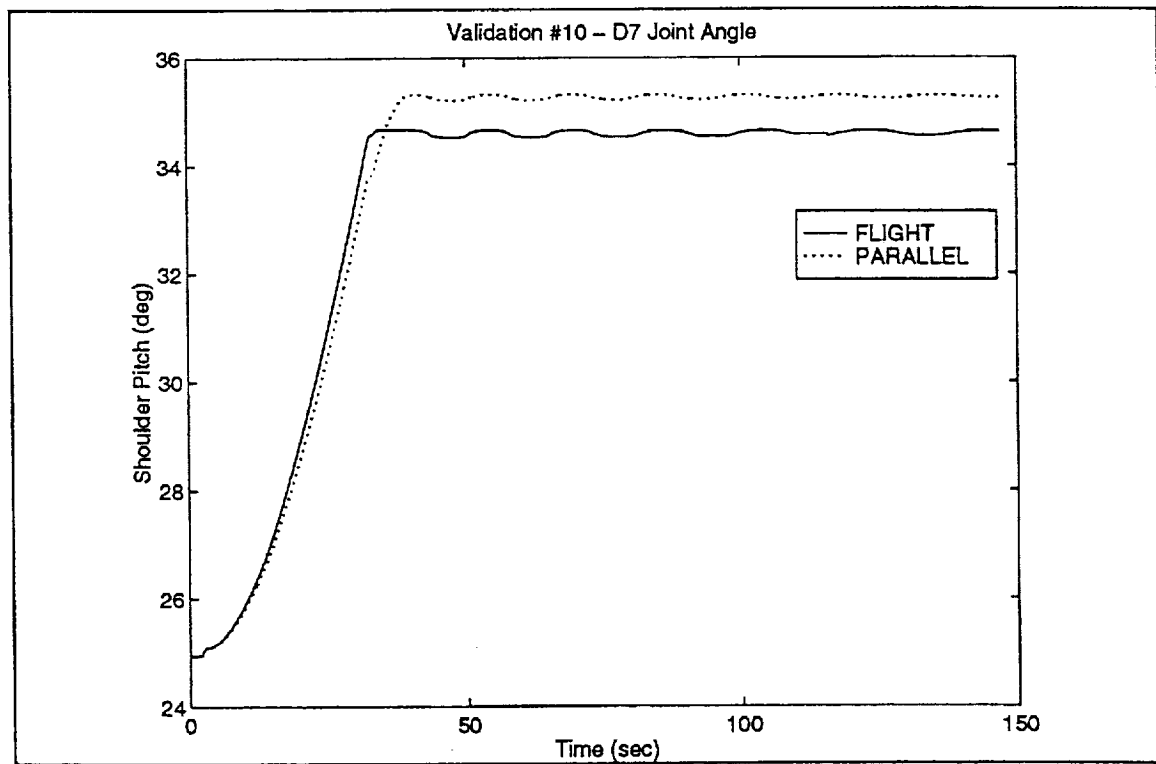


Figure B2: Validation #10 - D7 Joint Angle

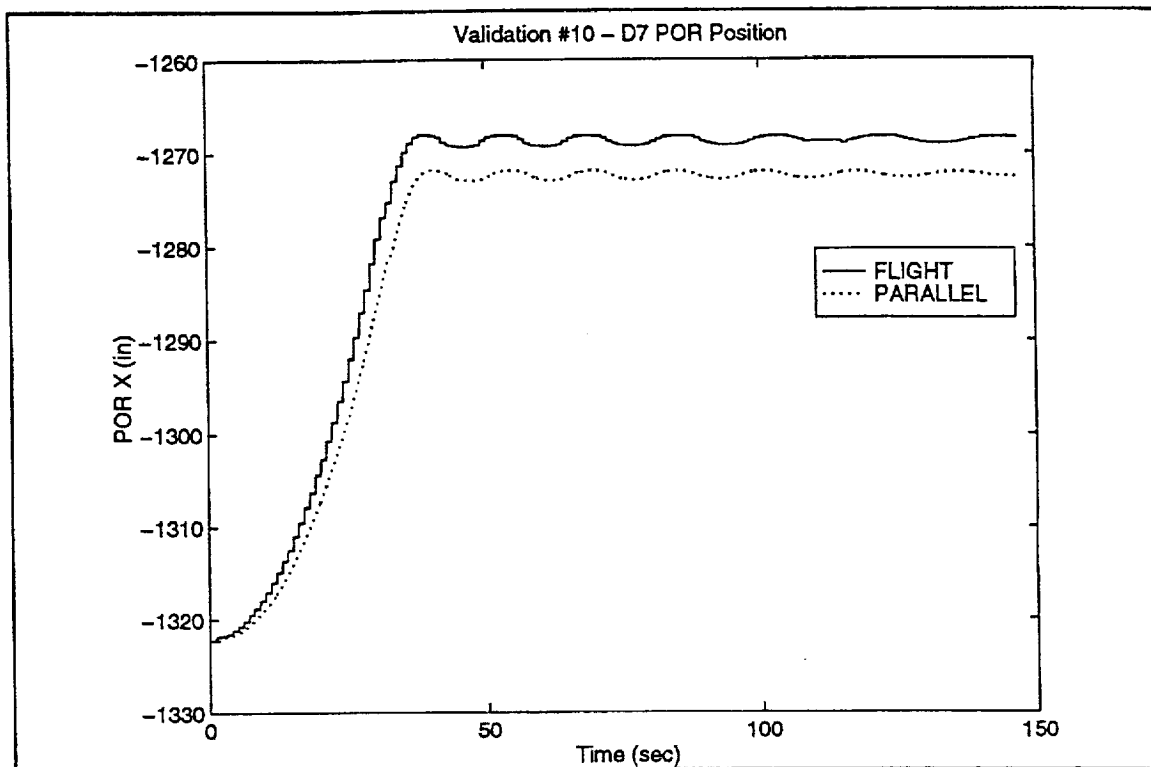


Figure B3: Validation #10 - D7 POR Position

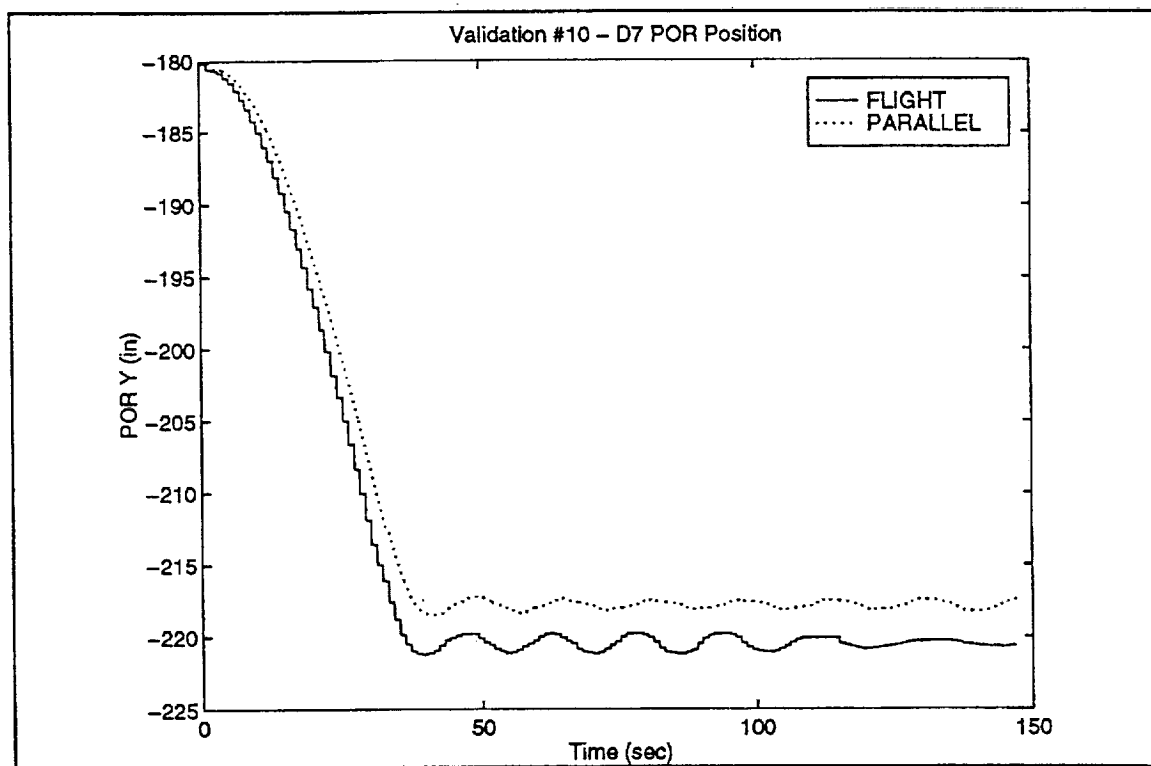


Figure B4: Validation #10 - D7 POR Position

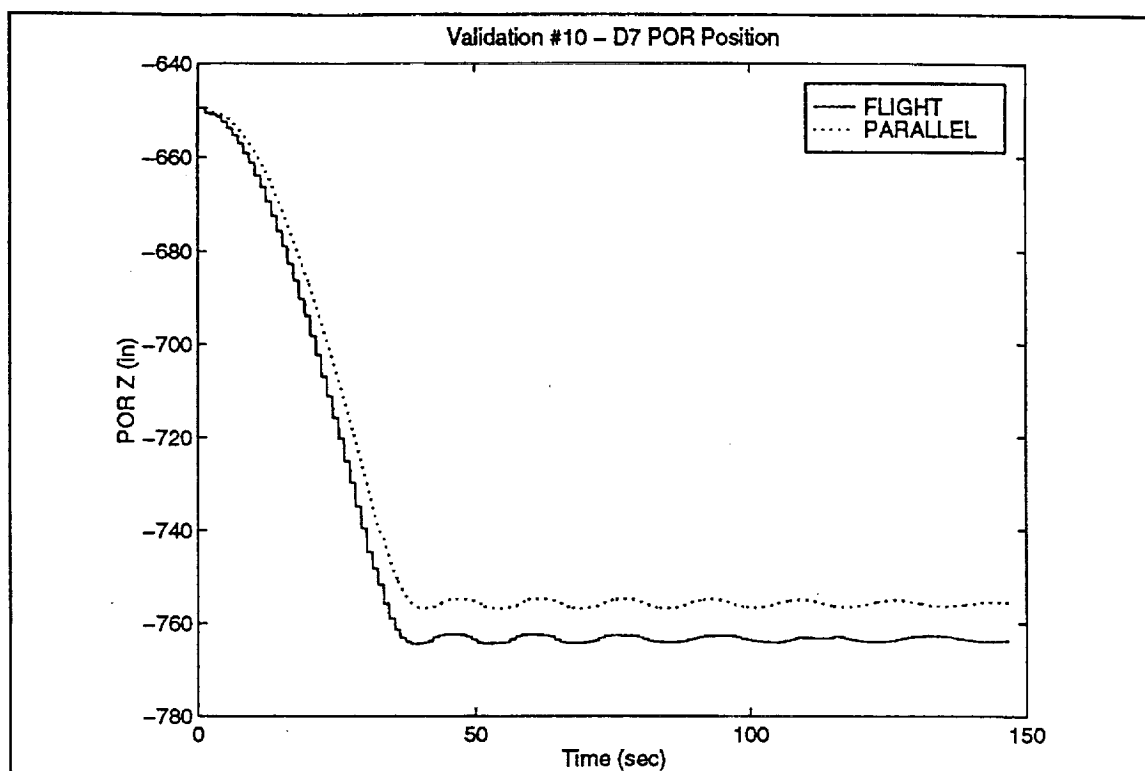


Figure B5: Validation #10 - D7 POR Position

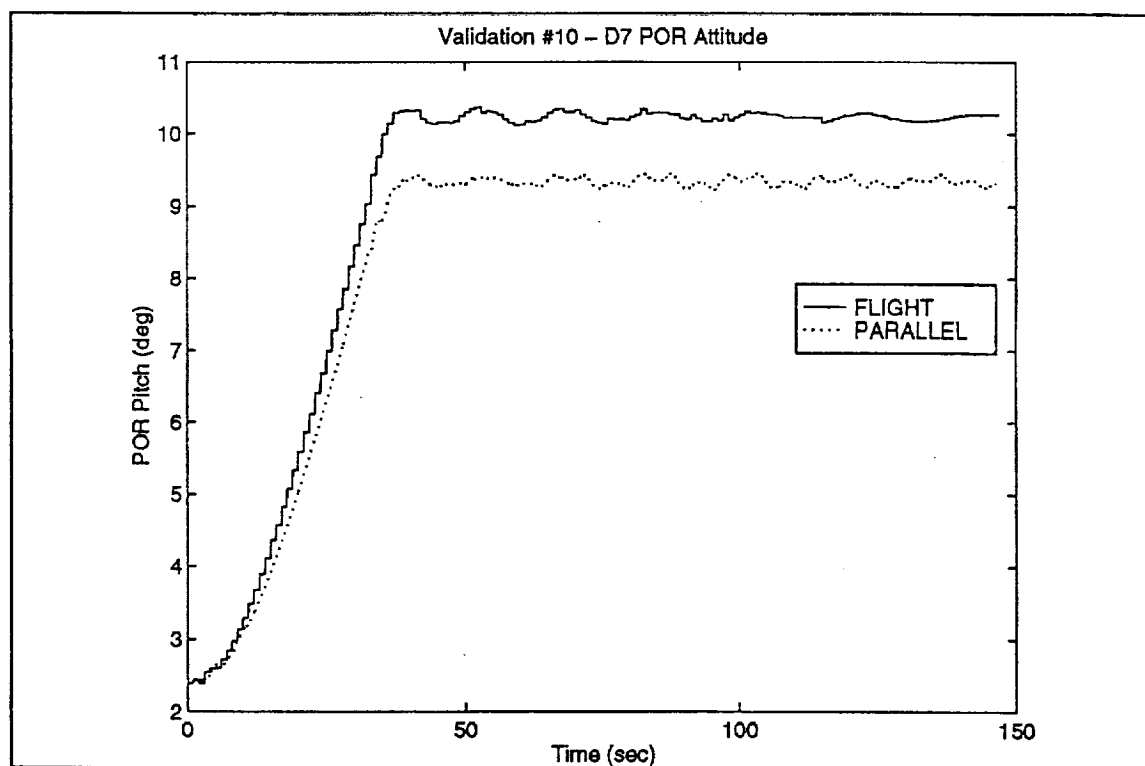


Figure B6: Validation #10 - D7 POR Attitude

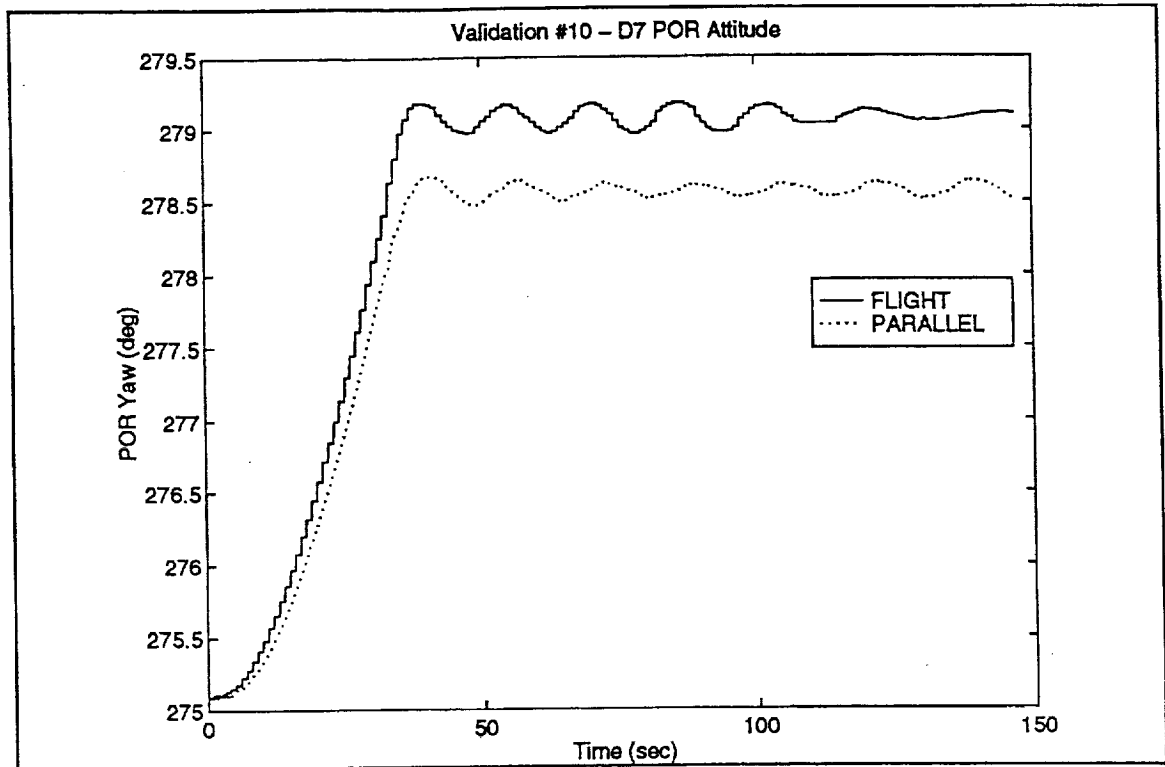


Figure B7: Validation #10 - D7 POR Attitude

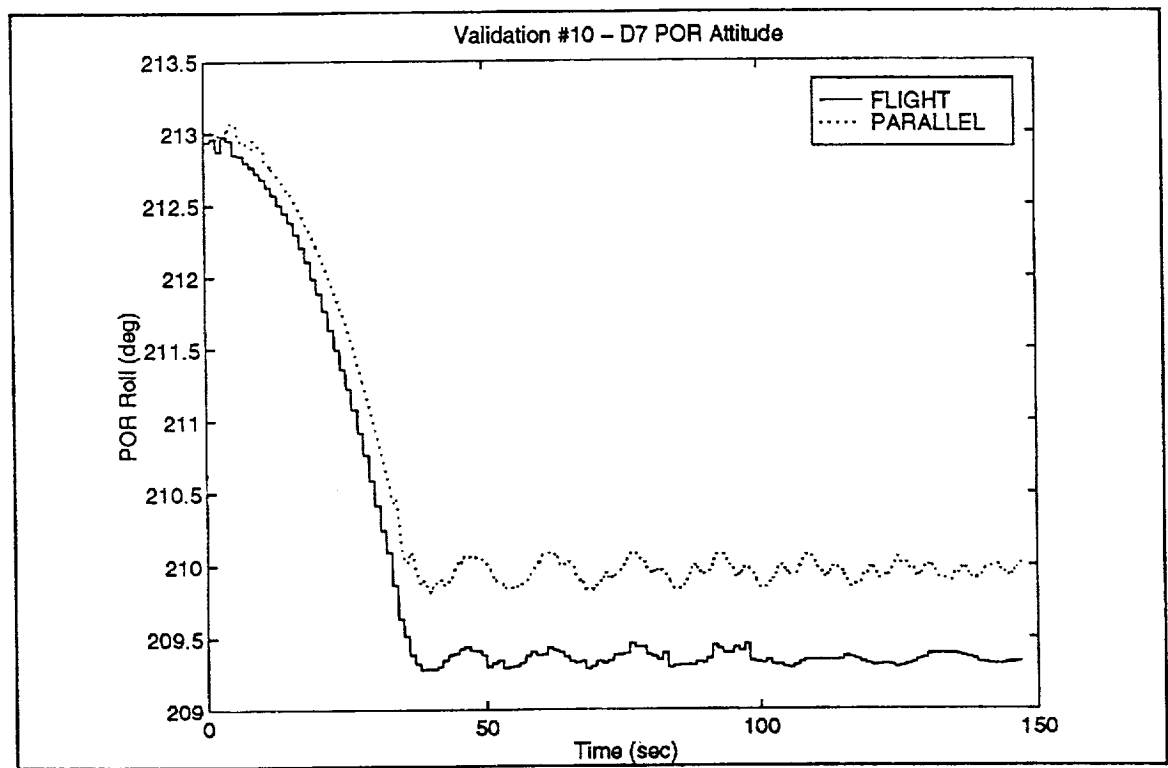


Figure B8: Validation #10 - D7 POR Attitude

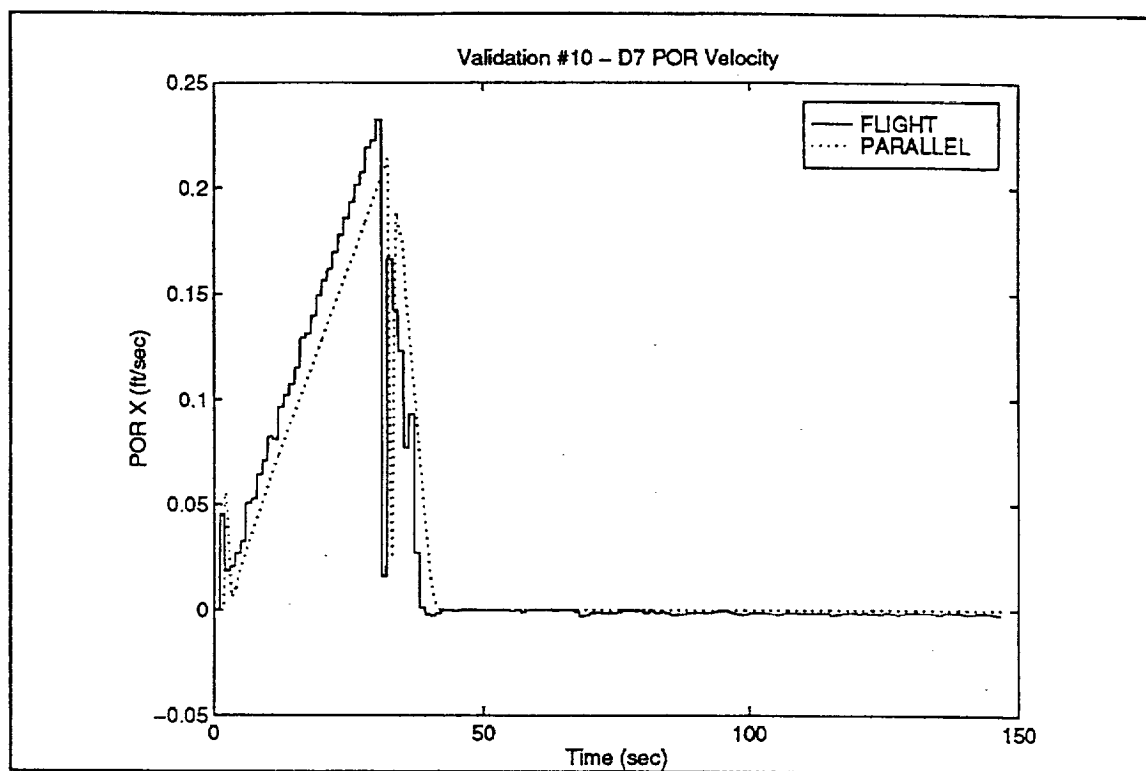


Figure B9: Validation #10 - D7 POR Velocity

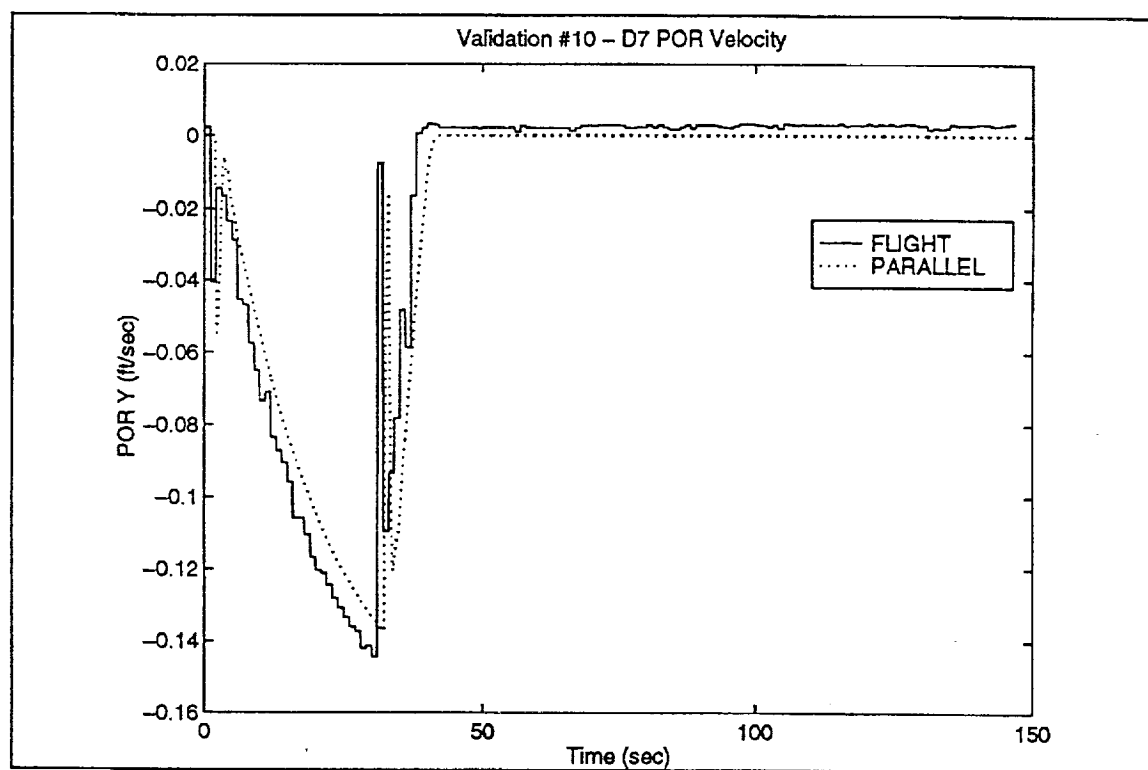


Figure B10: Validation #10 - D7 POR Velocity



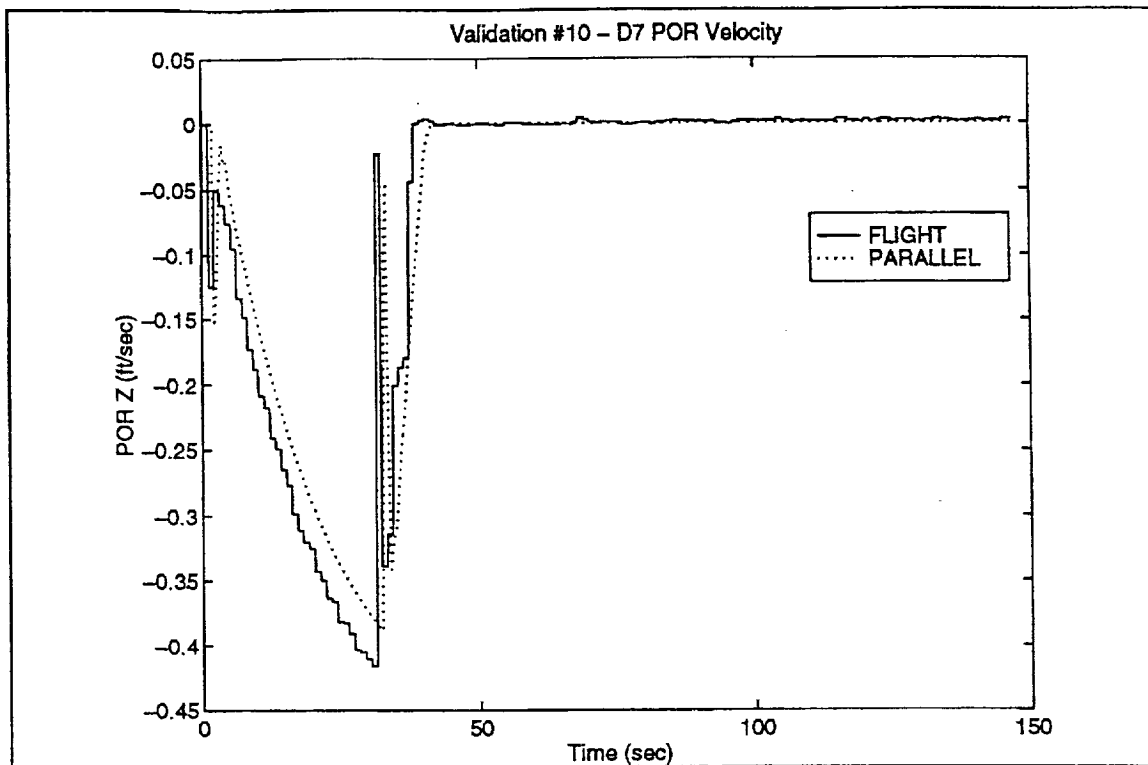


Figure B11: Validation #10 - D7 POR Velocity

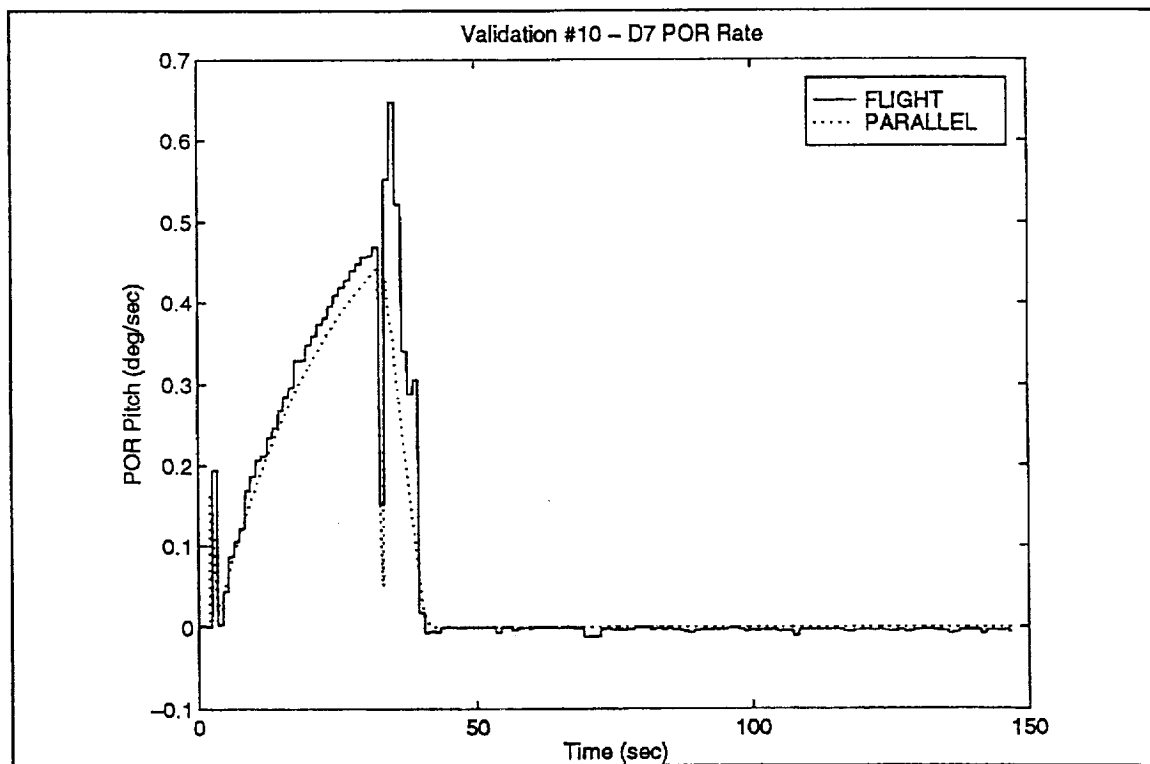


Figure B12: Validation #10 - D7 POR Rate

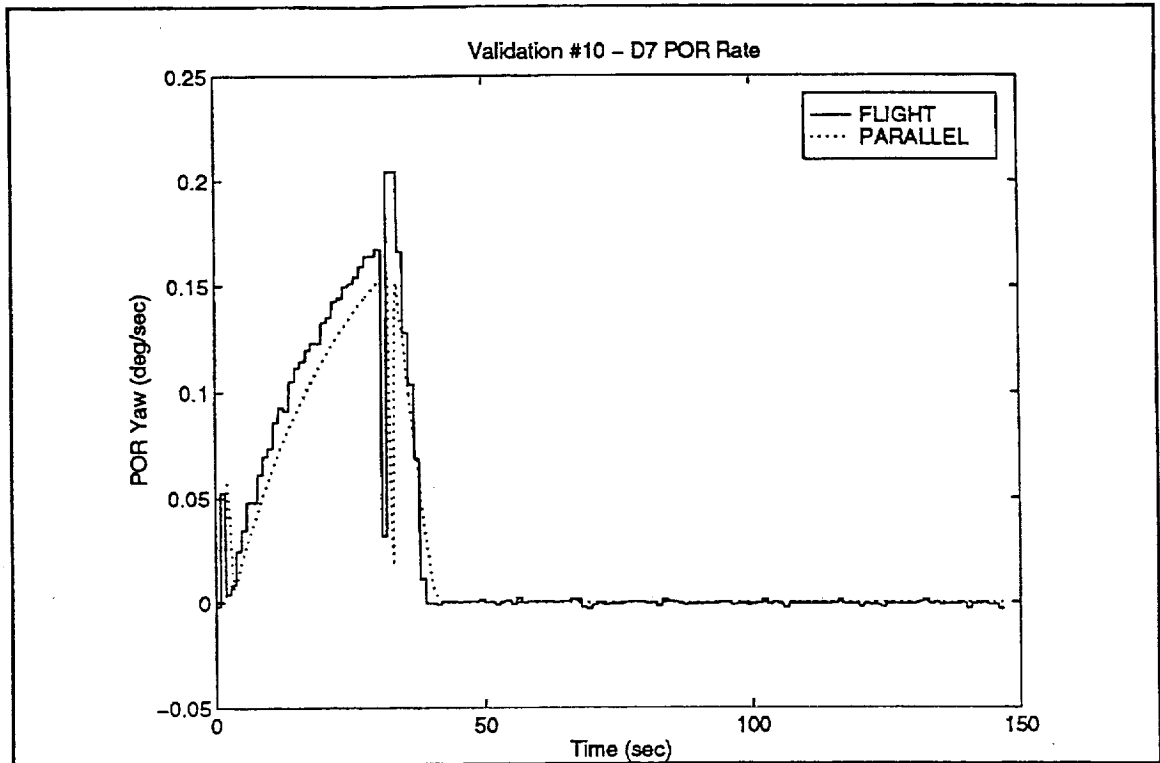


Figure B13: Validation #10 - D7 POR Rate

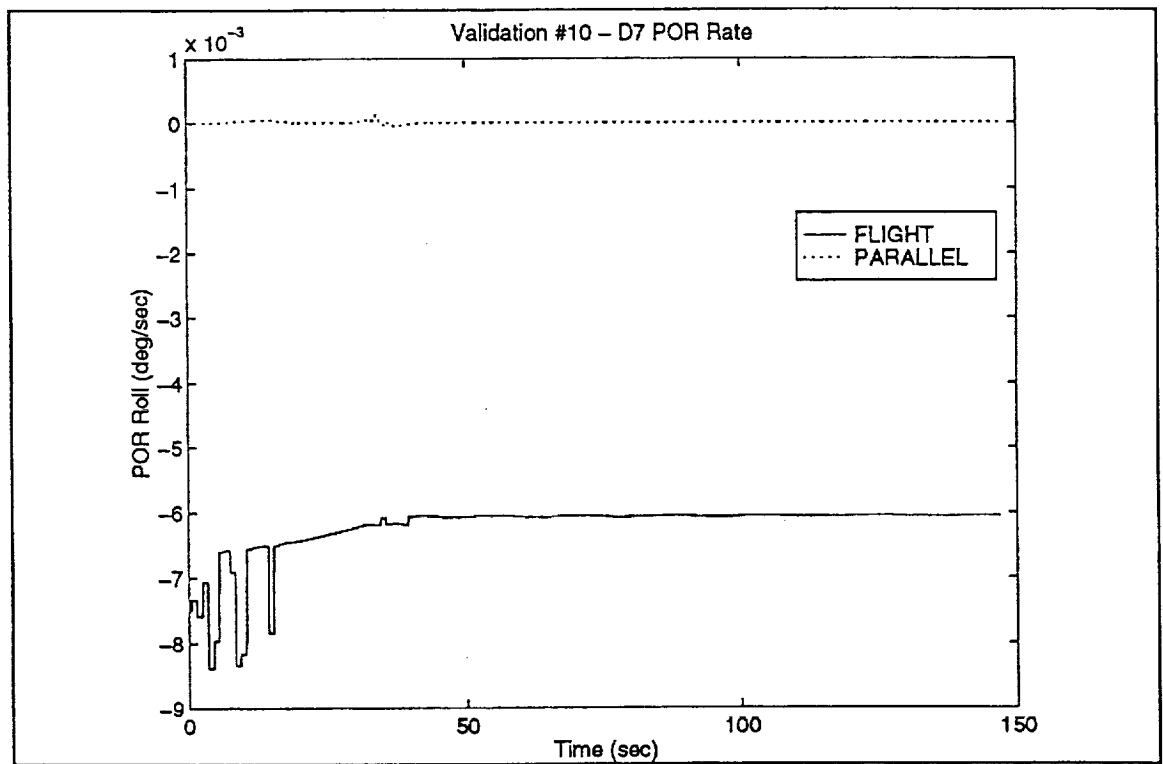


Figure B14: Validation #10 - D7 POR Rate

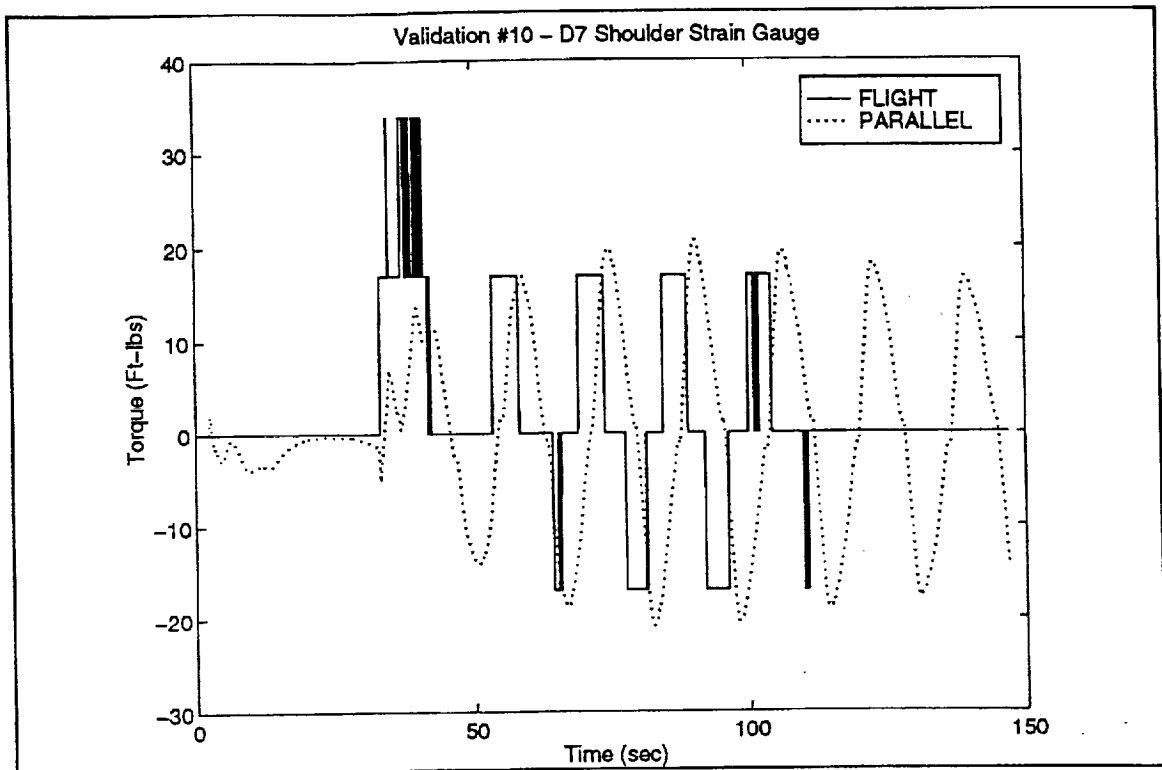


Figure B15: Validation #10 - D7 Shoulder Strain Gauge

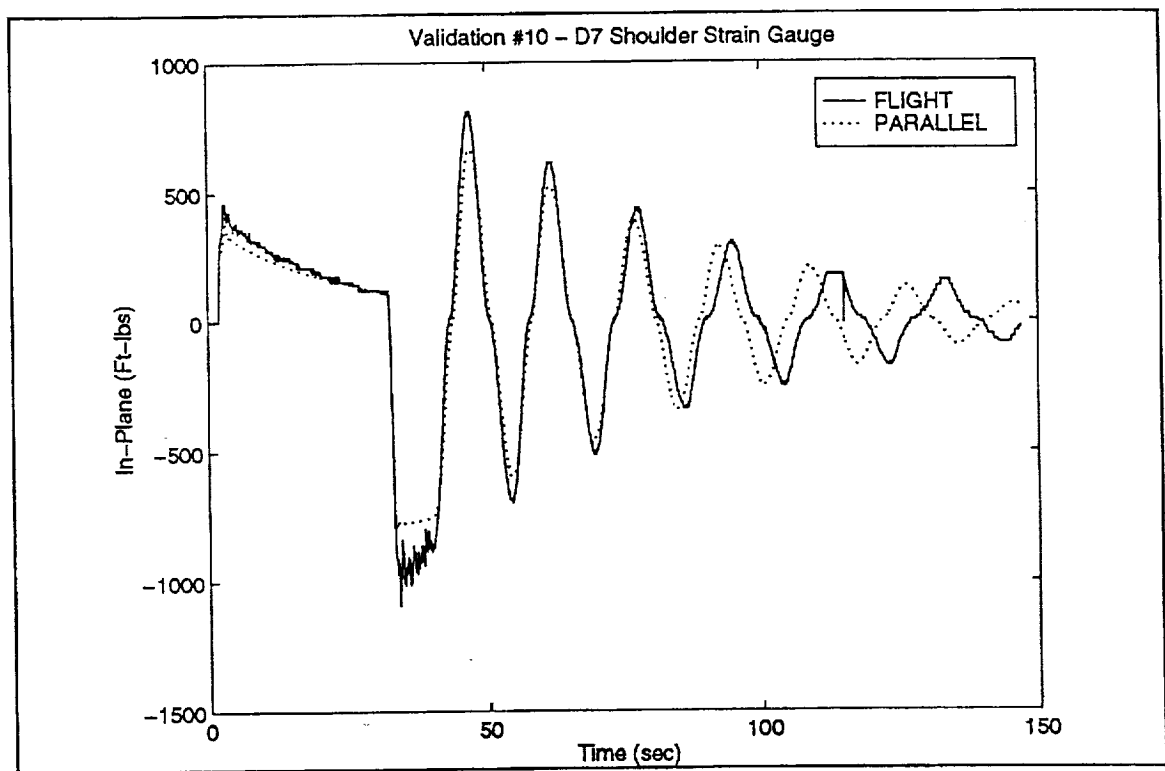


Figure B16: Validation #10 - D7 Shoulder Strain Gauge

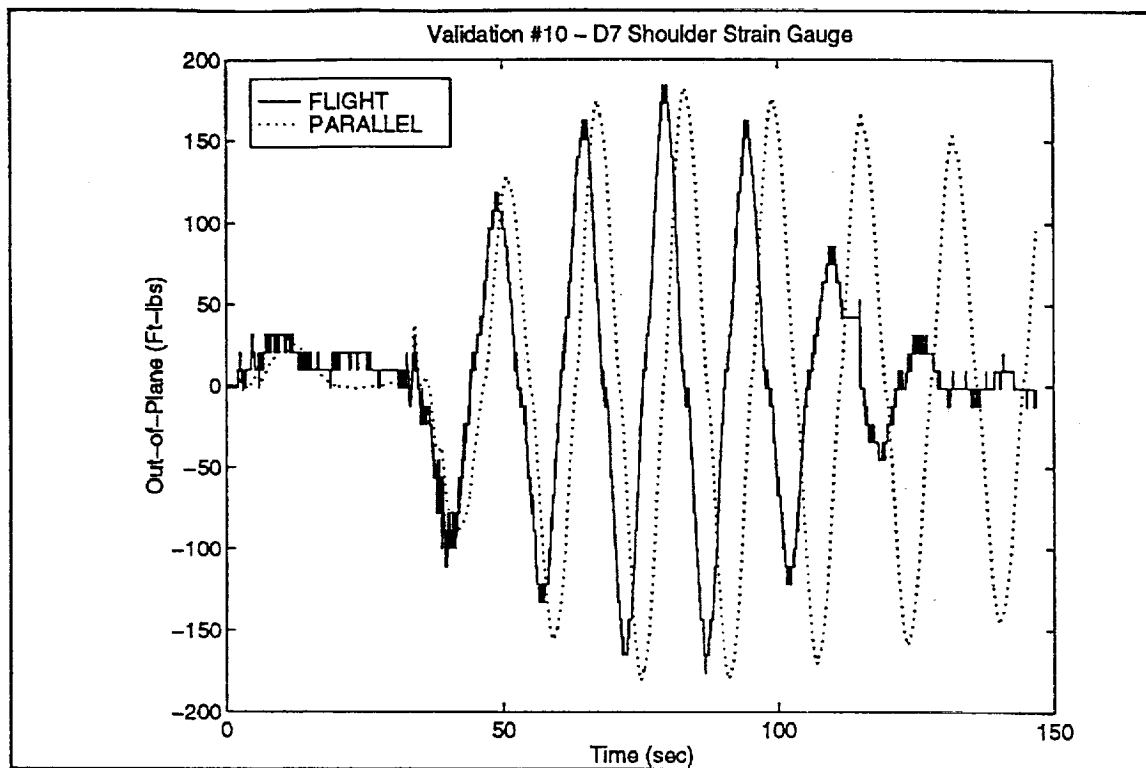


Figure B17: Validation #10 - D7 Shoulder Strain Gauge

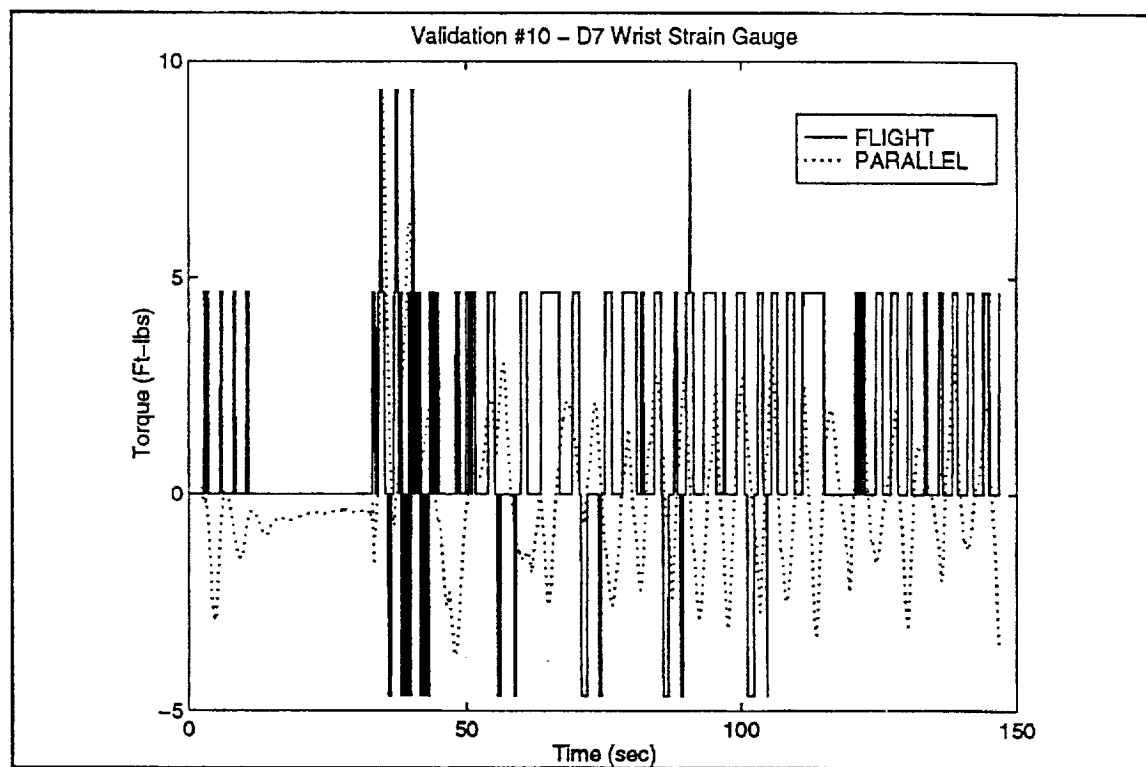


Figure B18: Validation #10 - D7 Wrist Strain Gauge

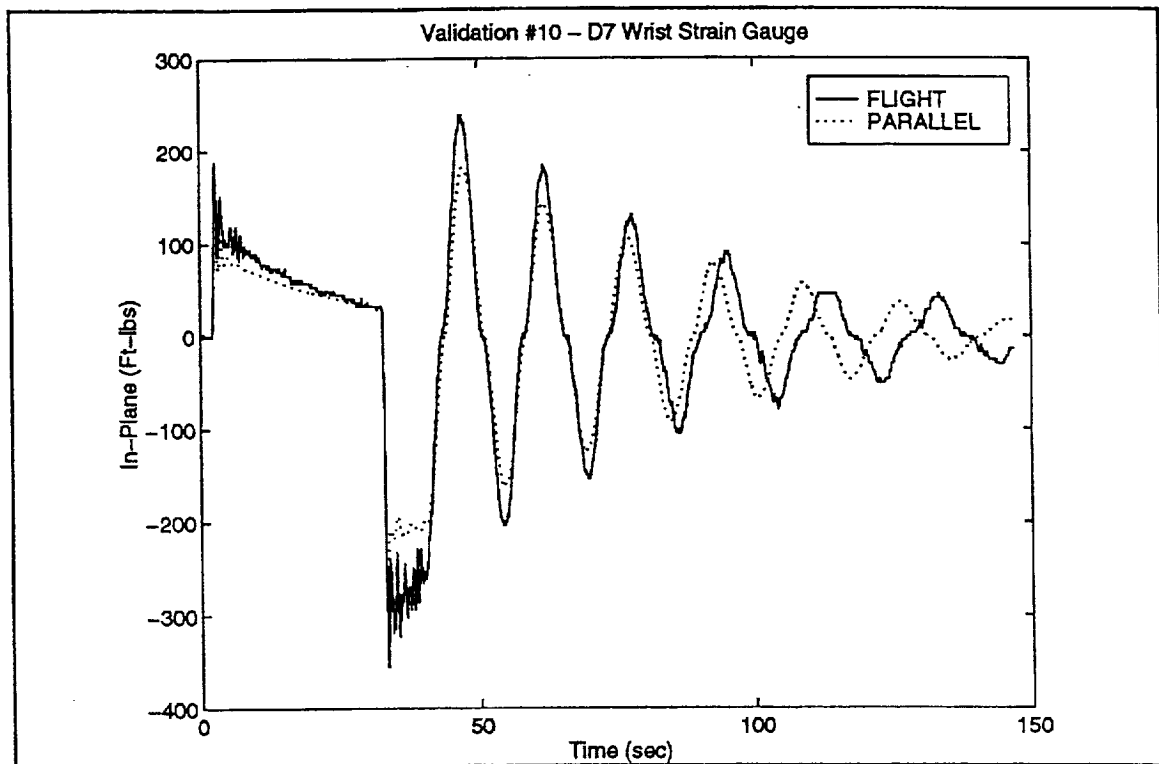


Figure B19: Validation #10 - D7 Wrist Strain Gauge

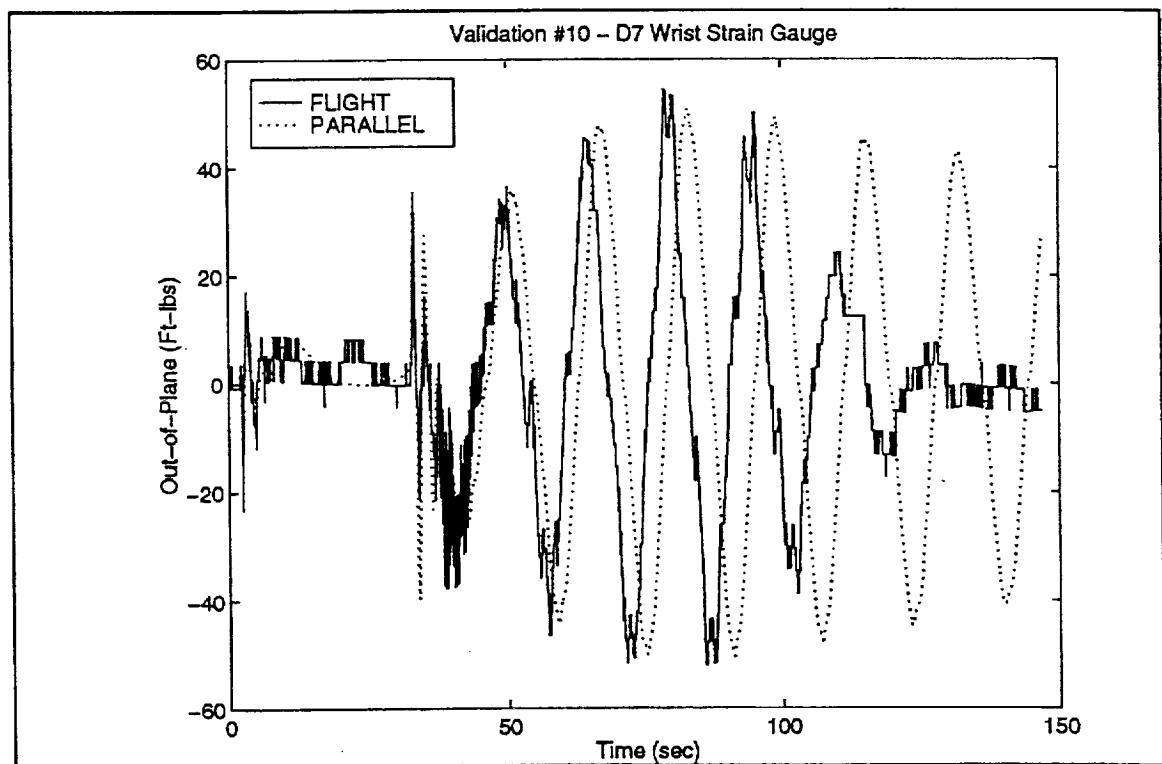


Figure B20: Validation #10 - D7 Wrist Strain Gauge

**APPENDIX C**  
**S7 SINGLE JOINT VALIDATION CASE**

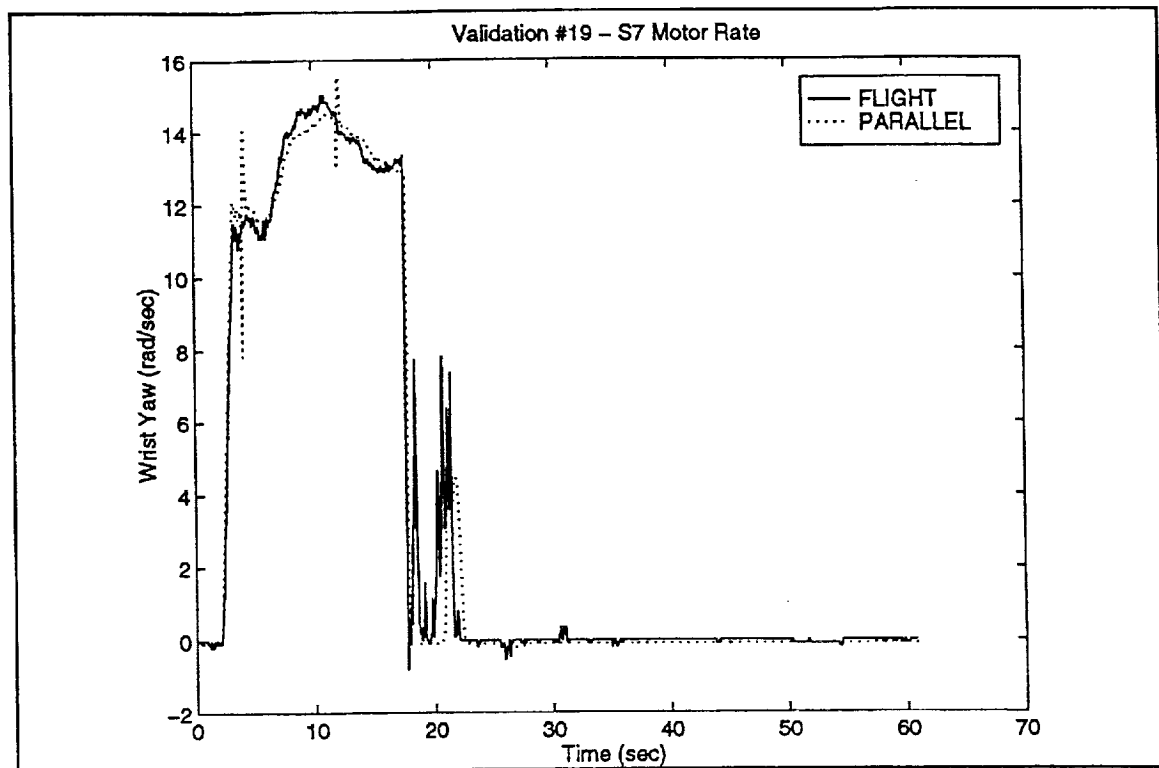


Figure C1: Validation #19 - S7 Motor Rate

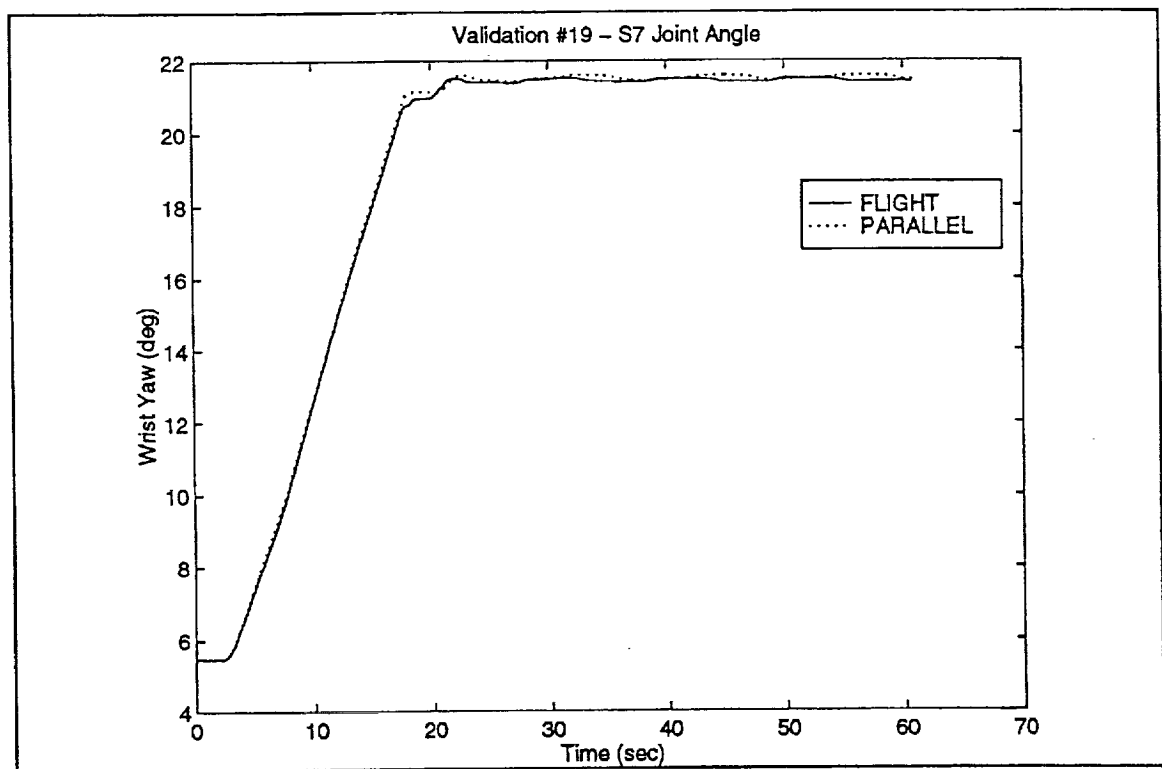


Figure C2: Validation #19 - S7 Joint Angle

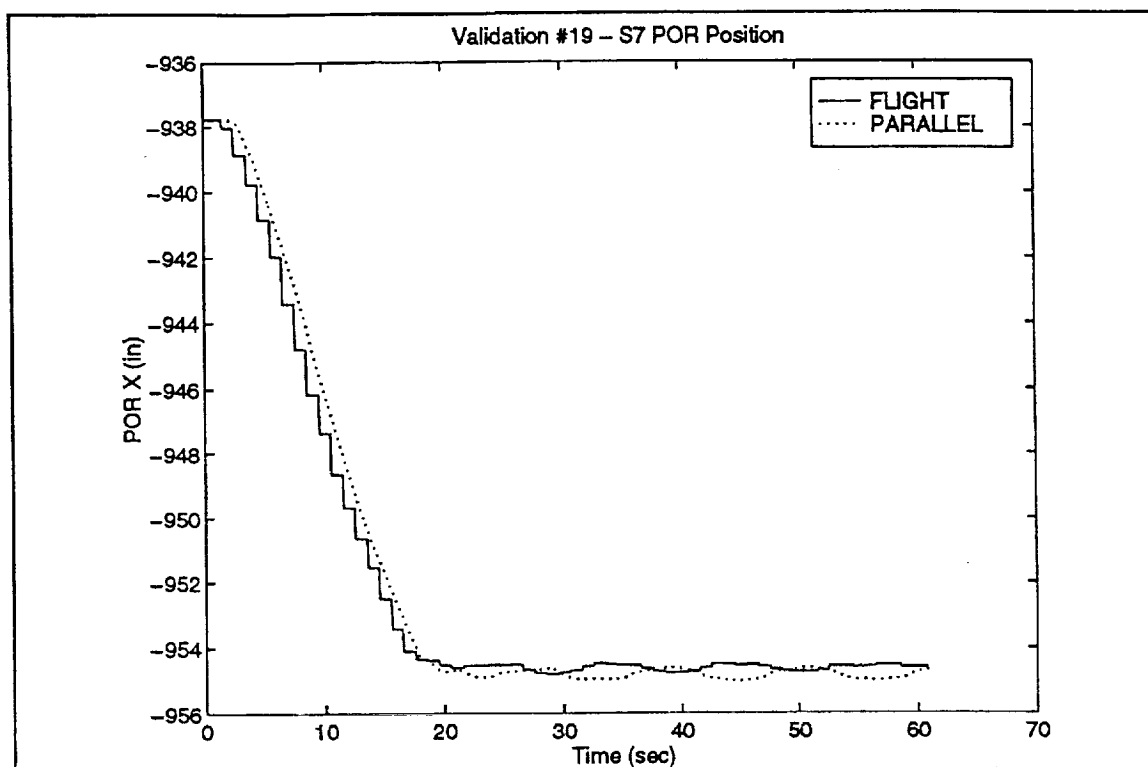


Figure C3: Validation #19 - S7 POR Position

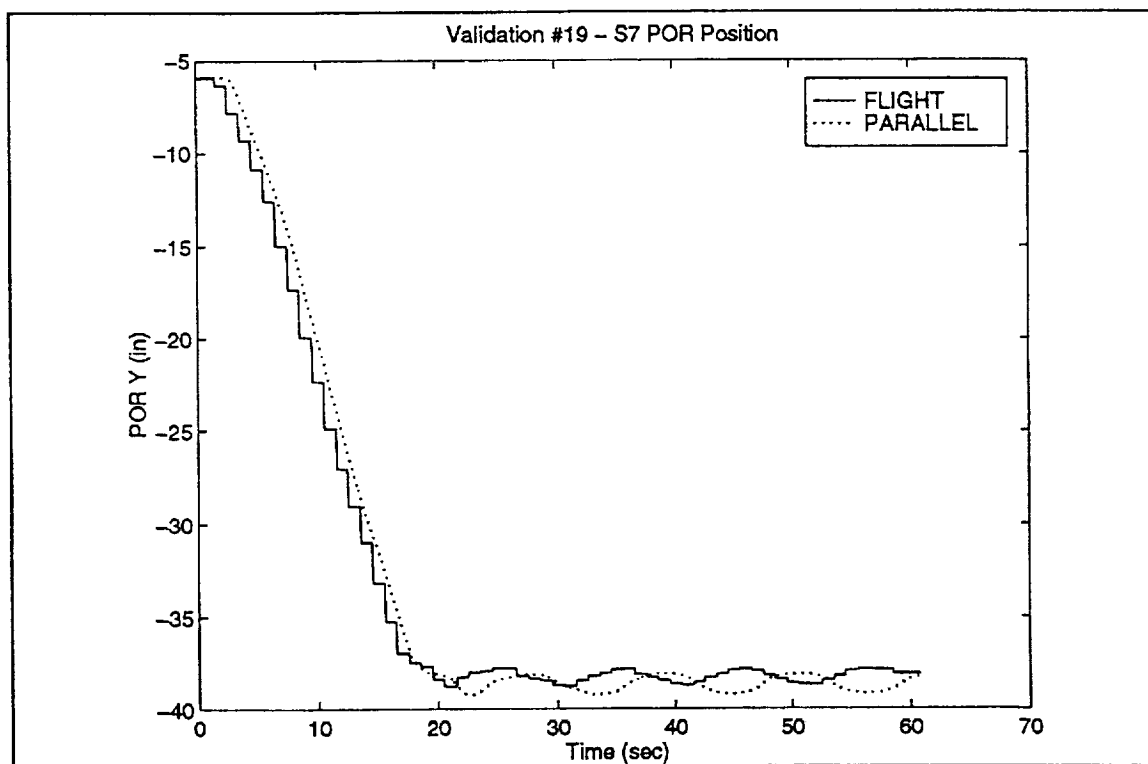


Figure C4: Validation #19 - S7 POR Position



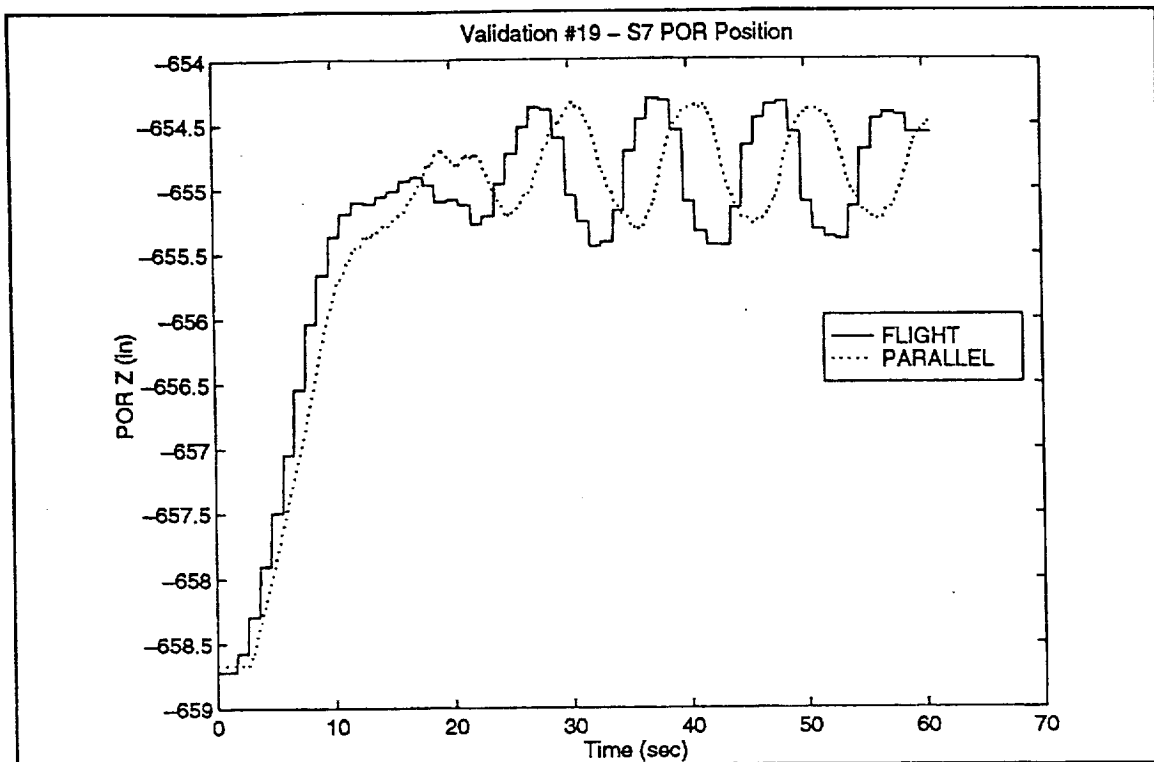


Figure C5: Validation #19 - D7 POR Position

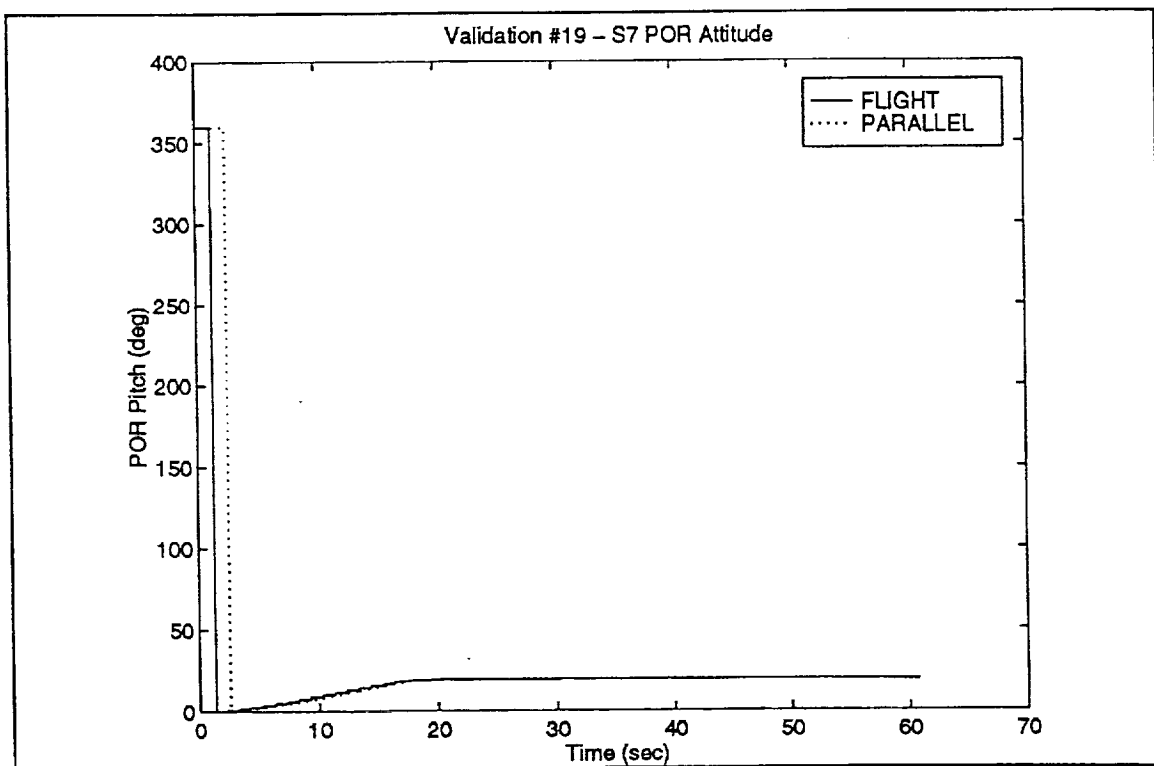


Figure C6: Validation #19 - S7 POR Attitude

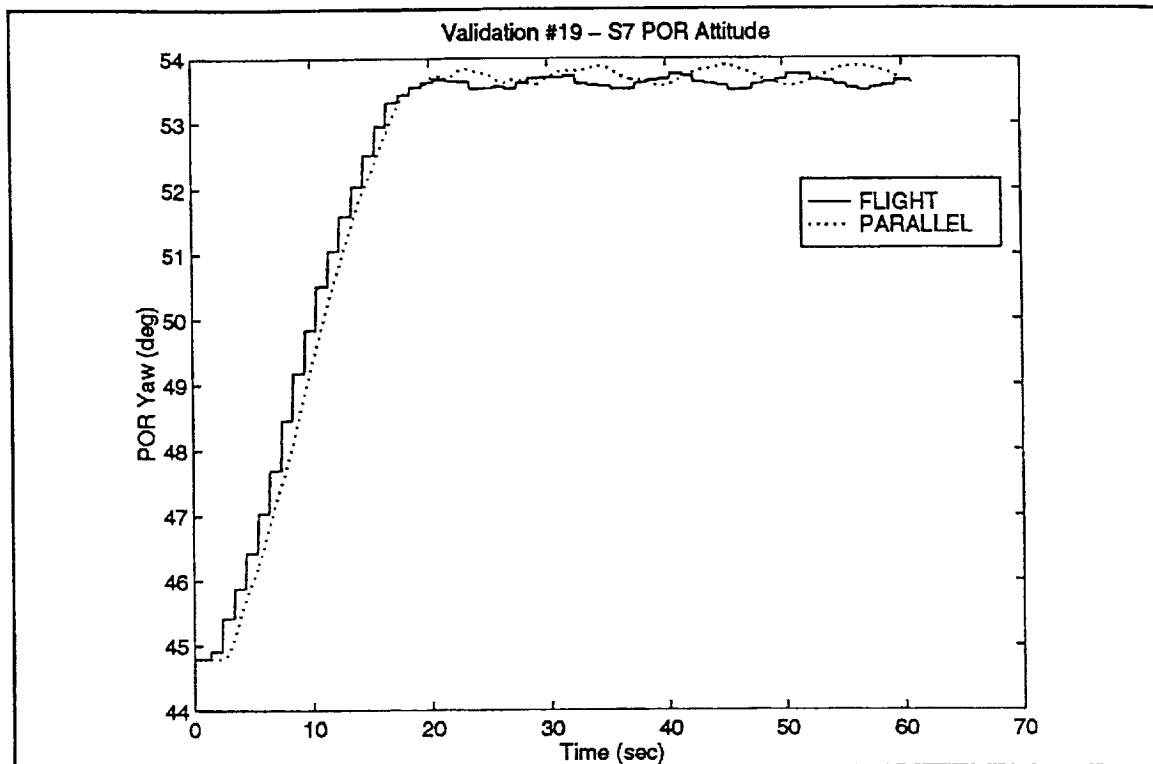


Figure C7: Validation #19 - S7 POR Attitude

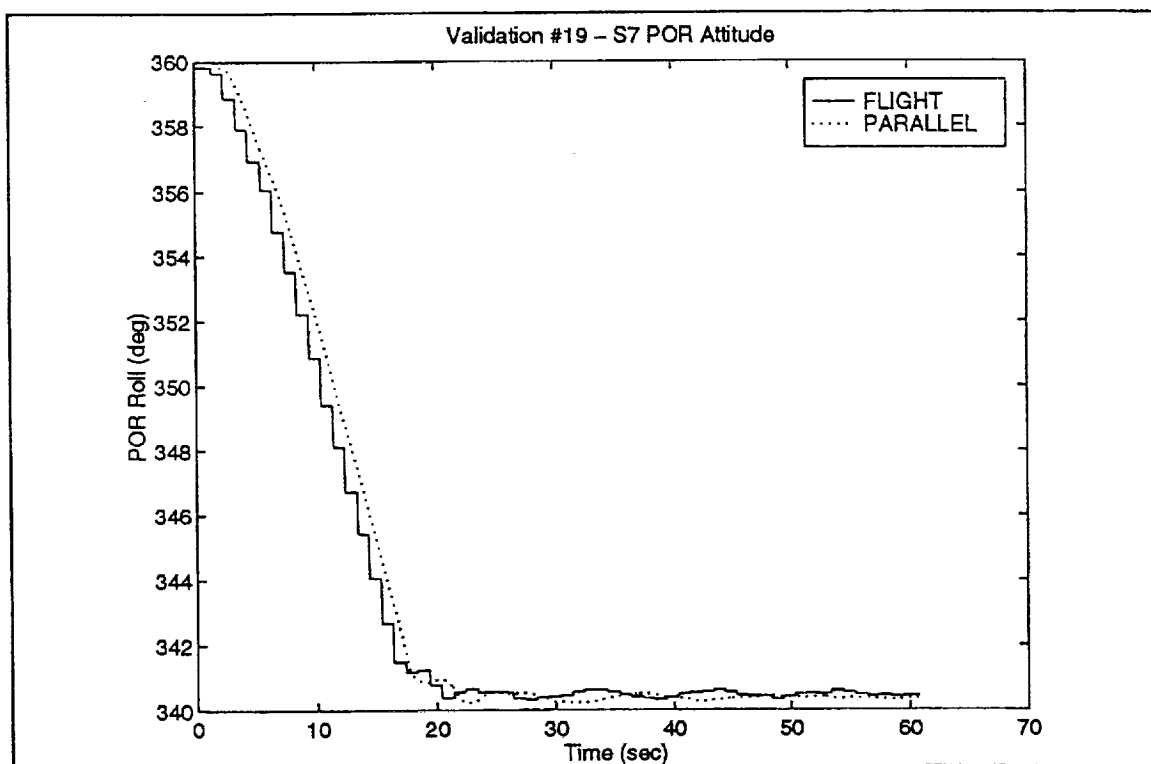


Figure C8: Validation #19 - S7 POR Attitude

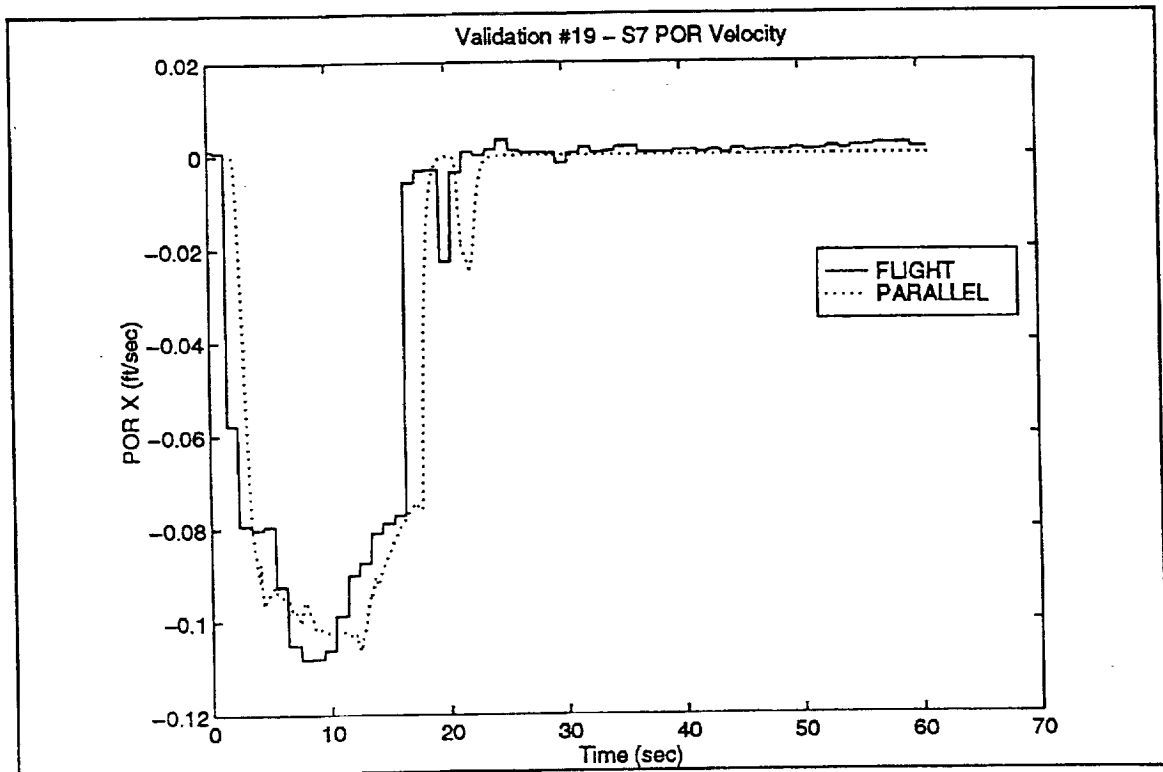


Figure C9: Validation #19 - S7 POR Velocity

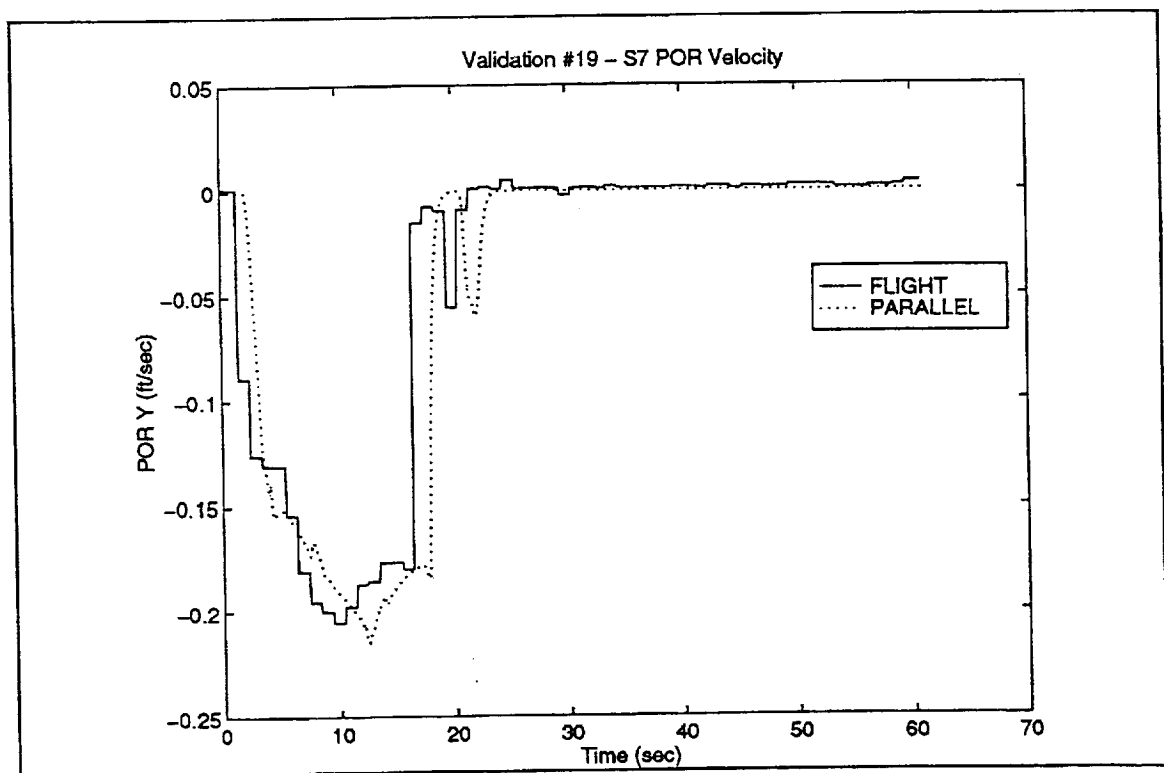


Figure C10: Validation #19 - S7 POR Velocity

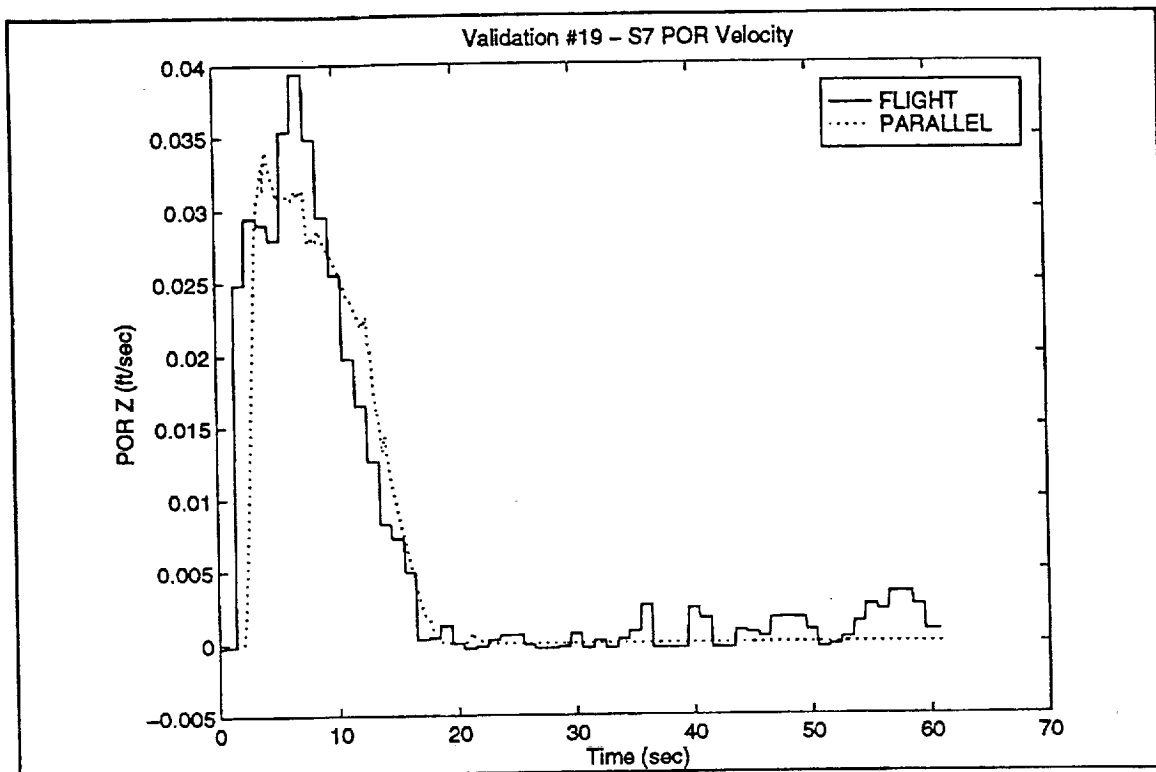


Figure C11: Validation #19 - S7 POR Velocity

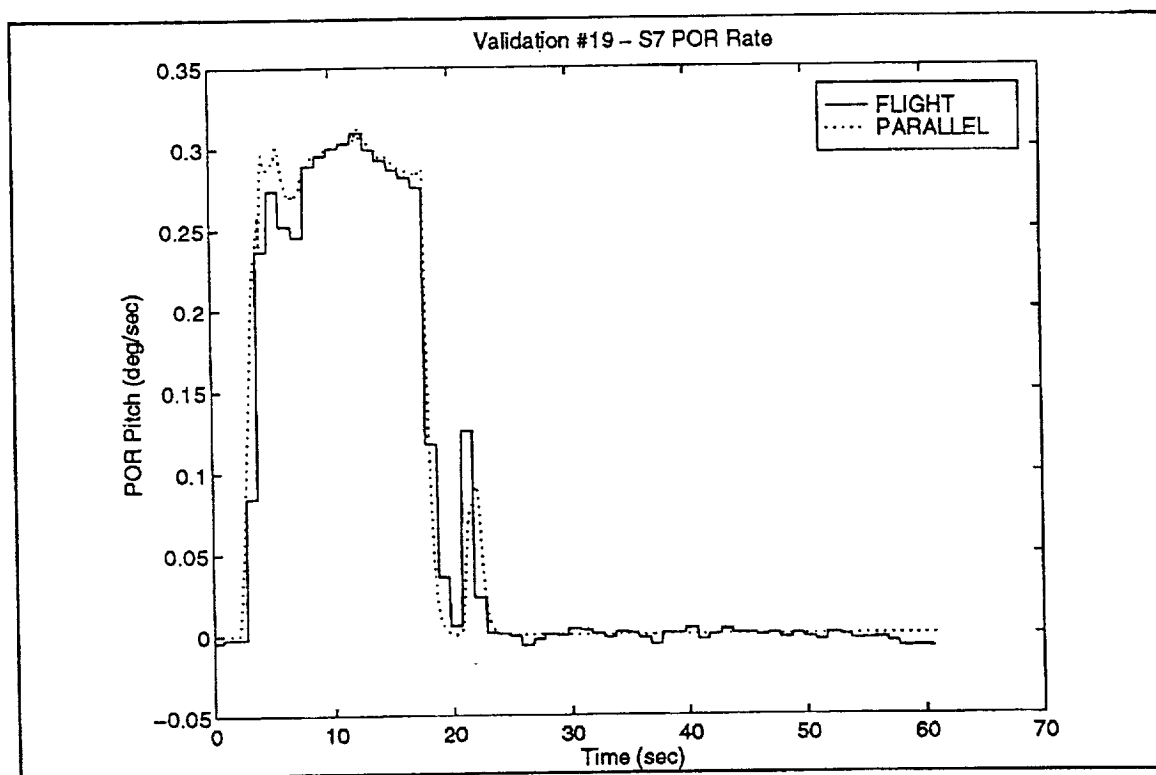


Figure C12: Validation #19 - S7 POR Rate

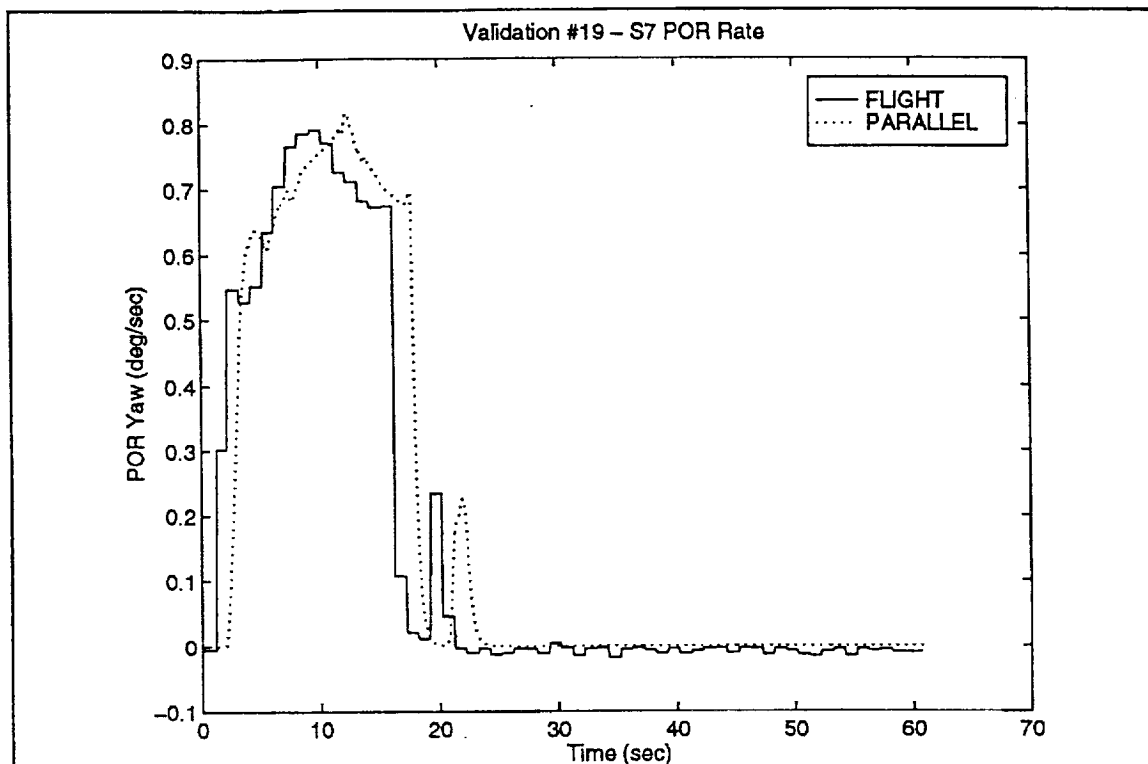


Figure C13: Validation #19 - S7 POR Rate

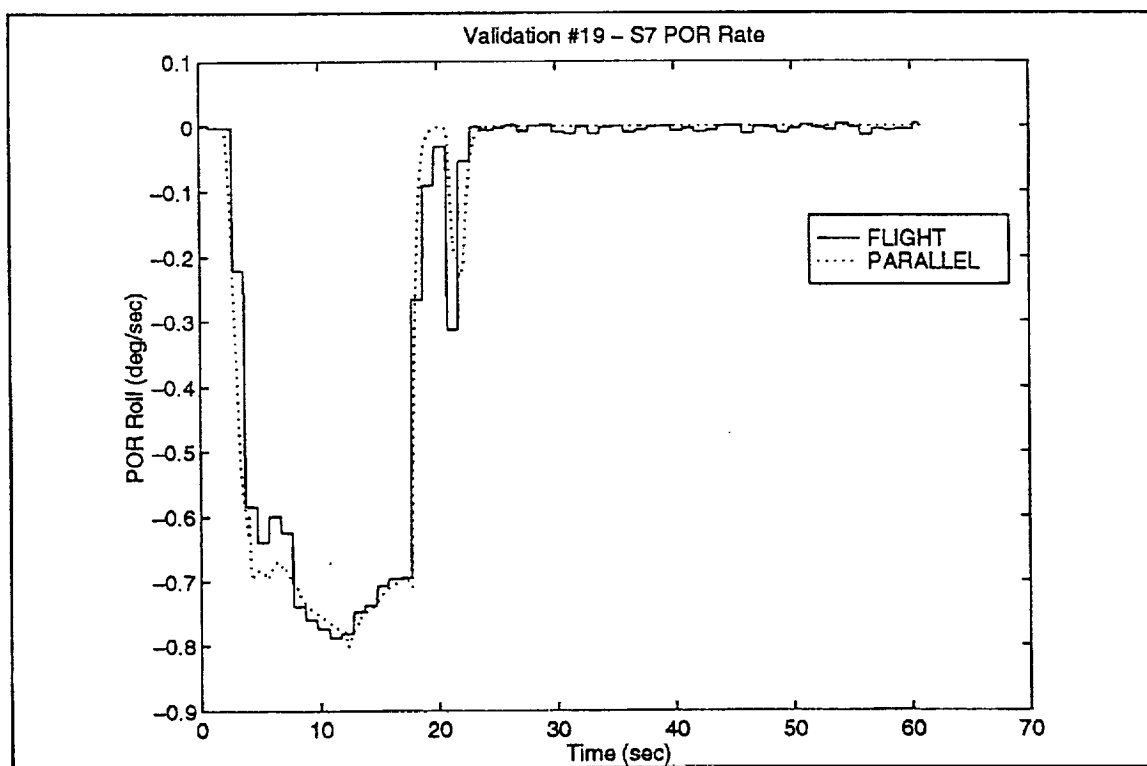


Figure C14: Validation #19 - S7 POR Rate

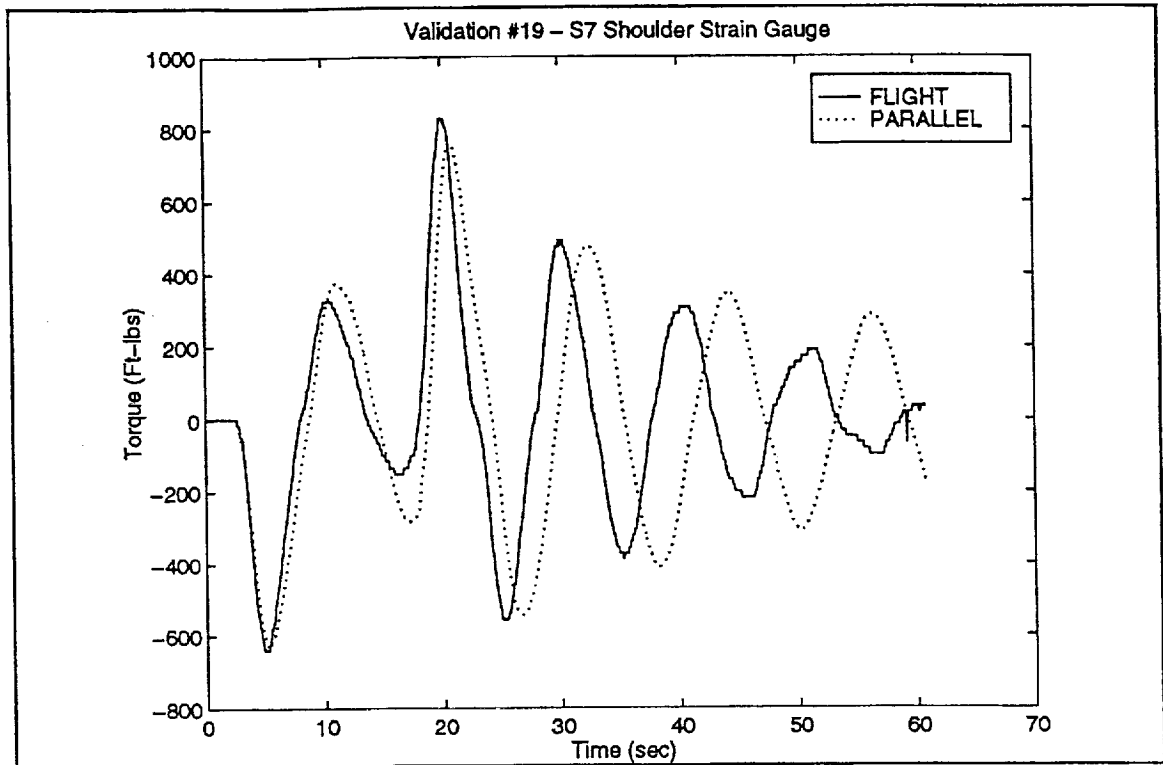


Figure C15: Validation #19 - S7 Shoulder Strain Gauge

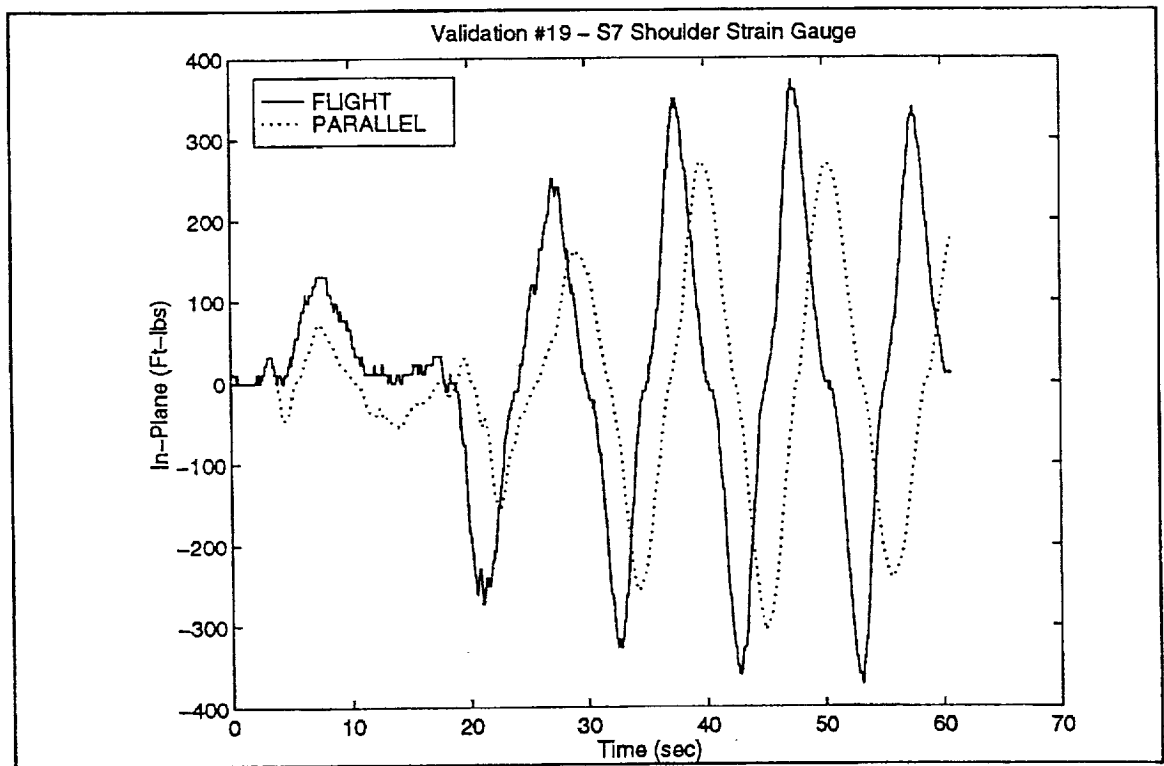


Figure C16: Validation #19 - S7 Shoulder Strain Gauge

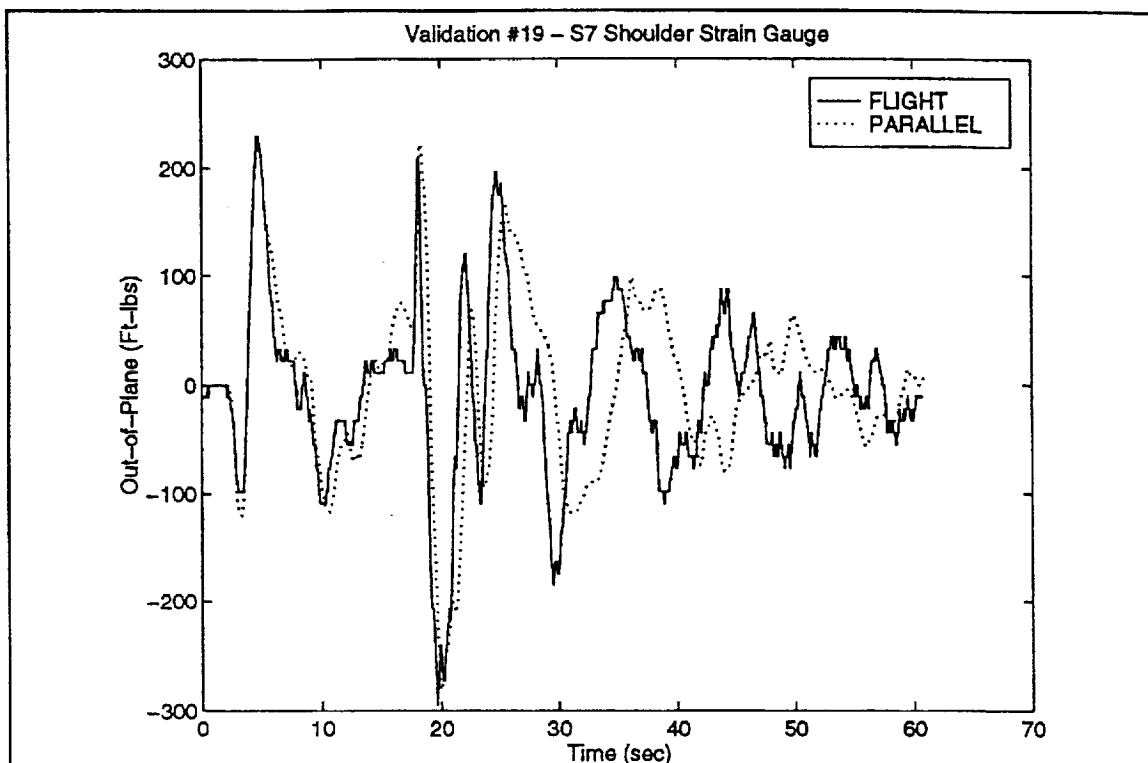


Figure C17: Validation #19 - S7 Shoulder Strain Gauge

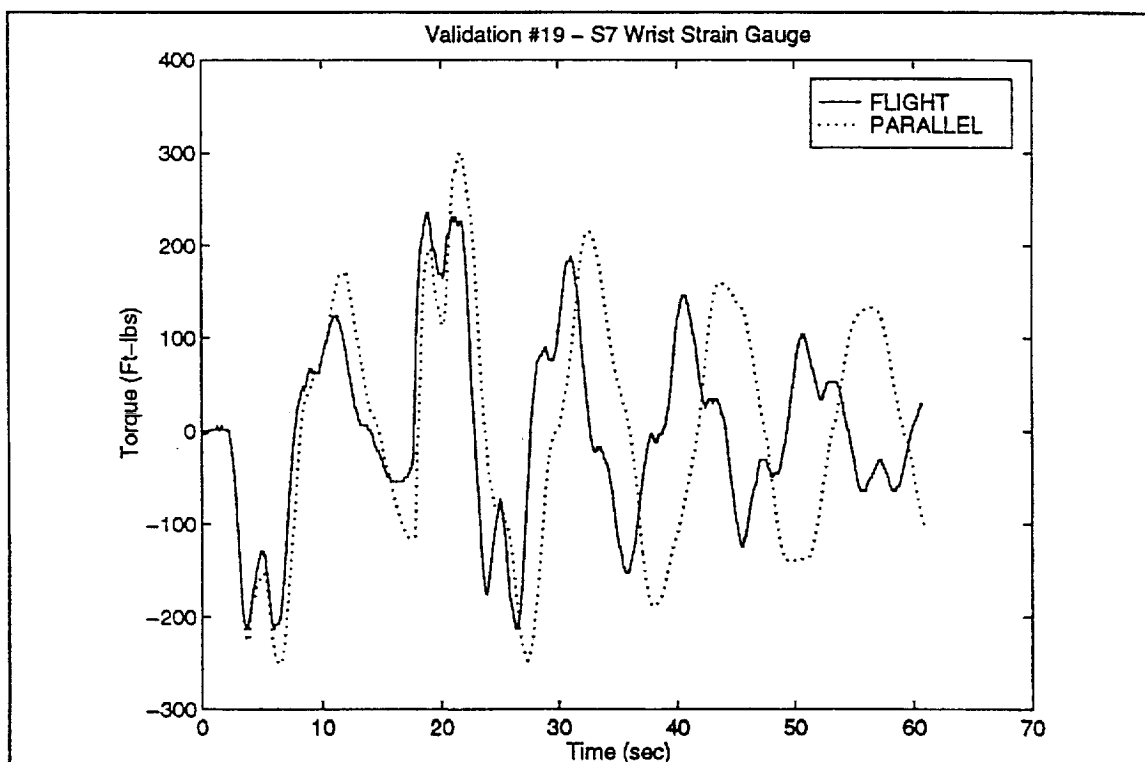


Figure C18: Validation #19 - S7 Wrist Strain Gauge

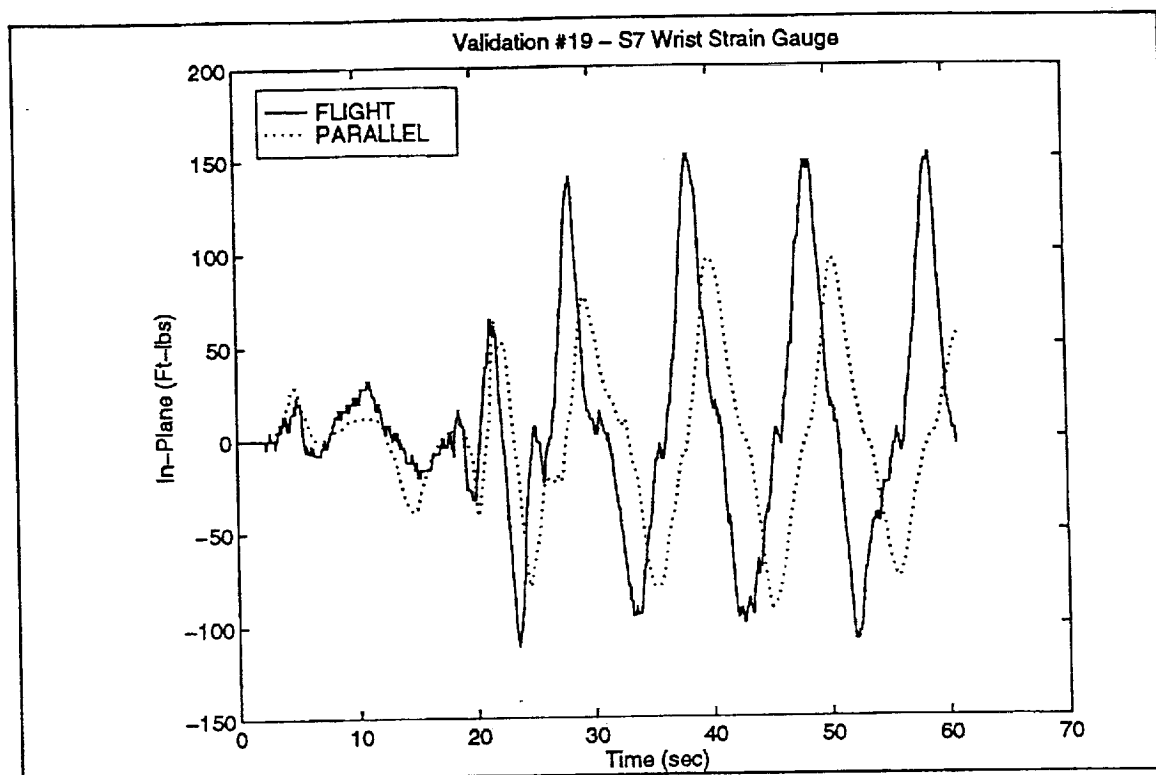


Figure C19: Validation #19 - S7 Wrist Strain Gauge

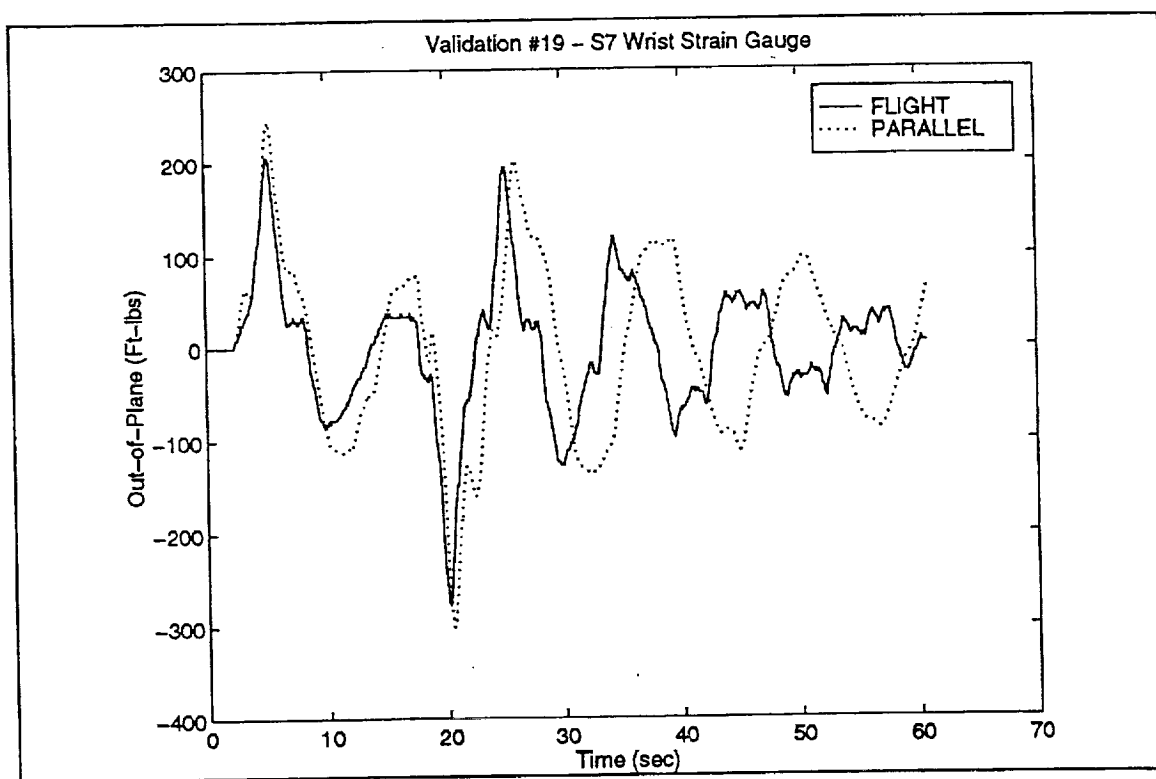


Figure C20: Validation #19 - S7 Wrist Strain Gauge



**APPENDIX D**  
**M5 MAN-IN-THE-LOOP VALIDATION CASE**

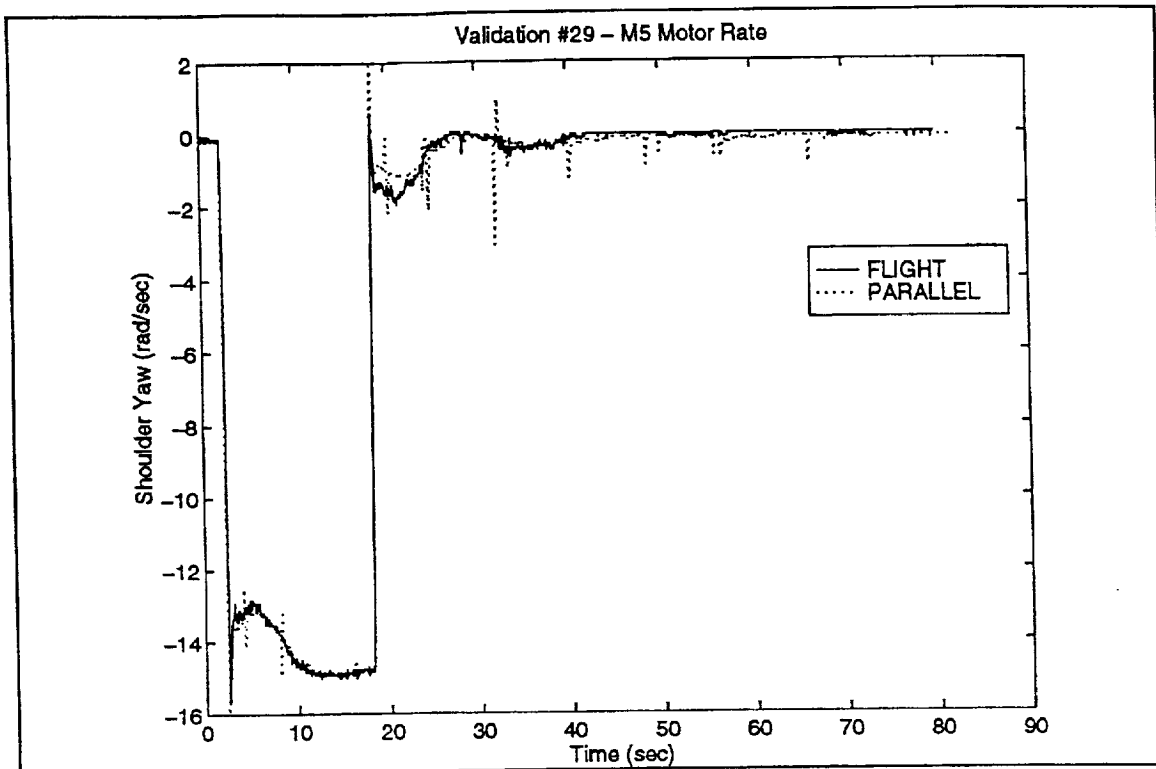


Figure D1: Validation #29 - M5 Motor Rate

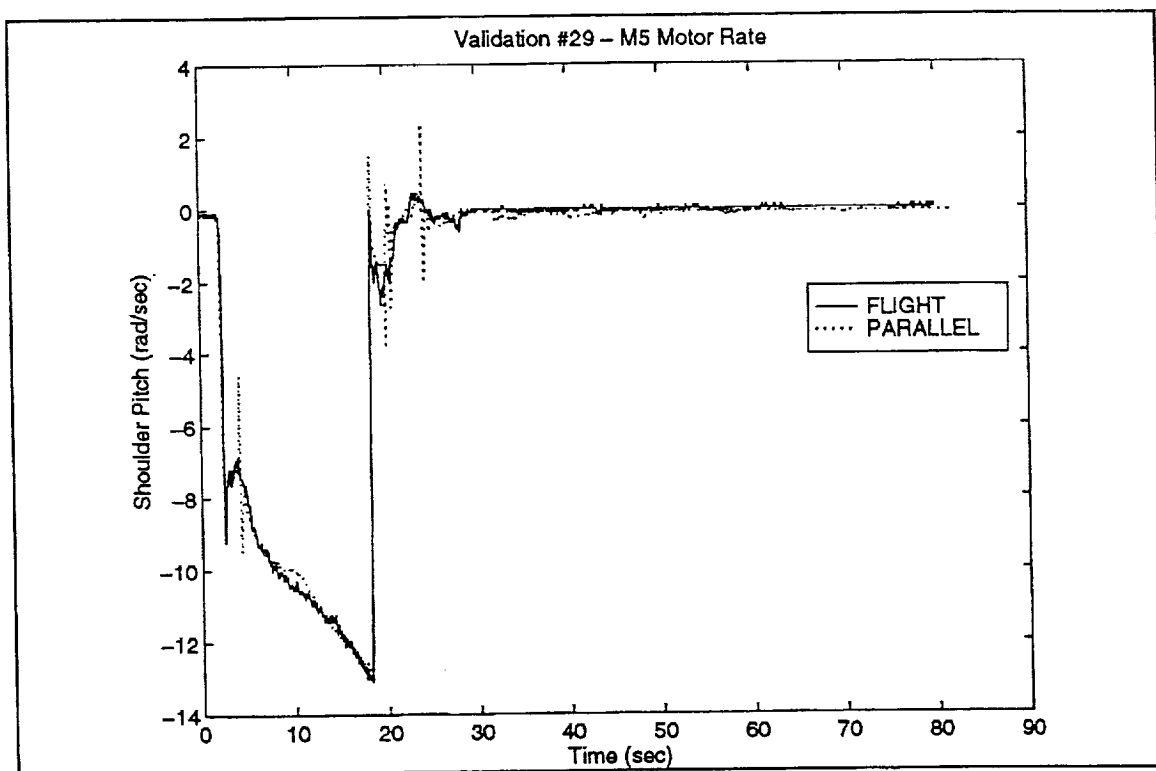


Figure D2: Validation #29 - M5 Motor Rate

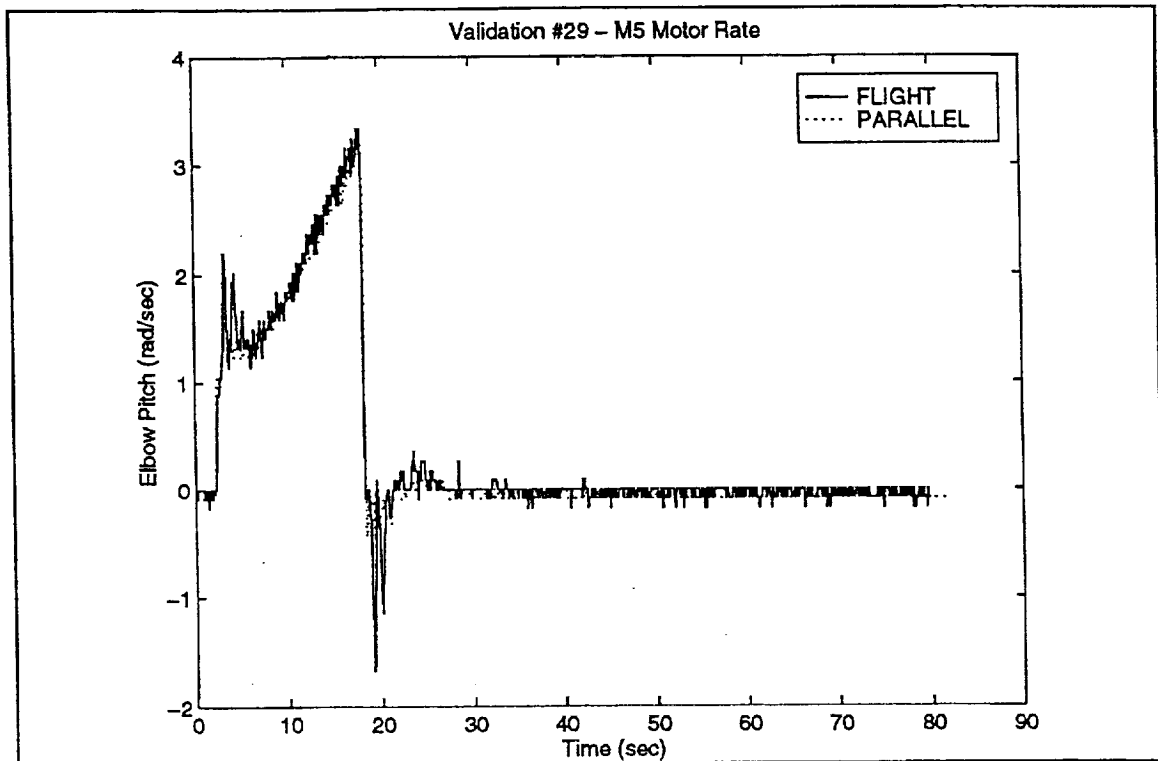


Figure D3: Validation #29 - M5 Motor Rate

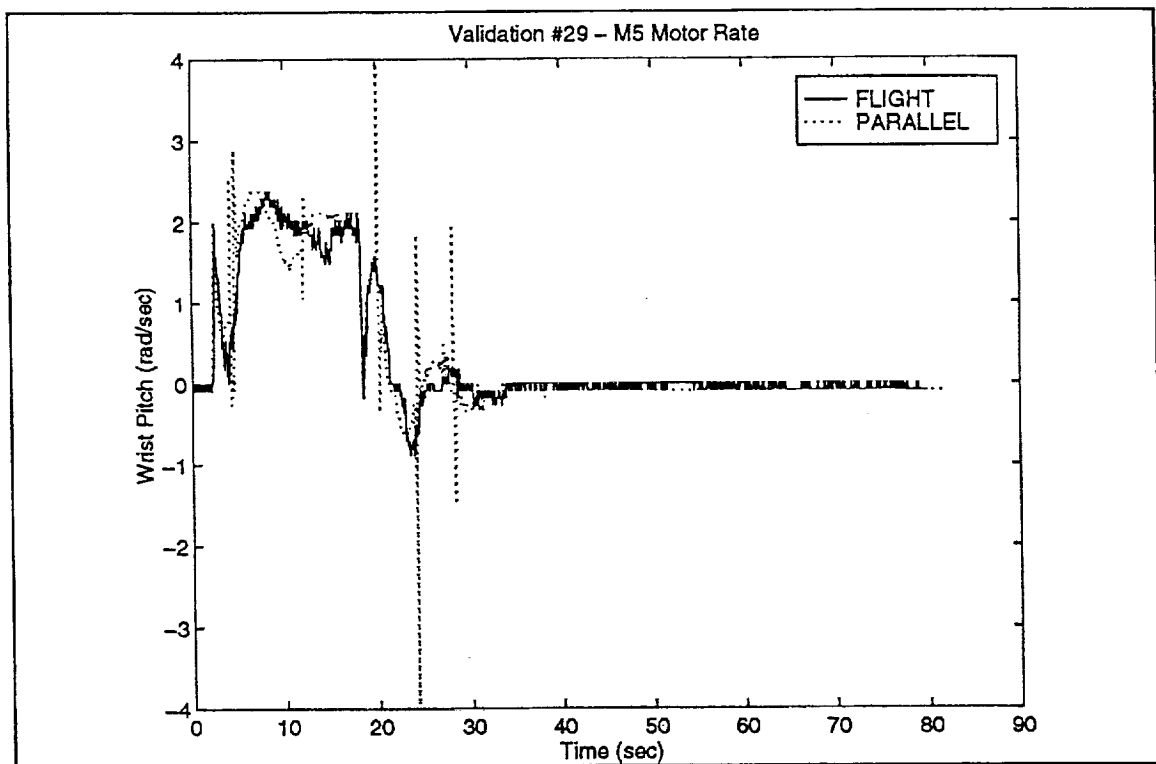


Figure D4: Validation #29 - M5 Motor Rate

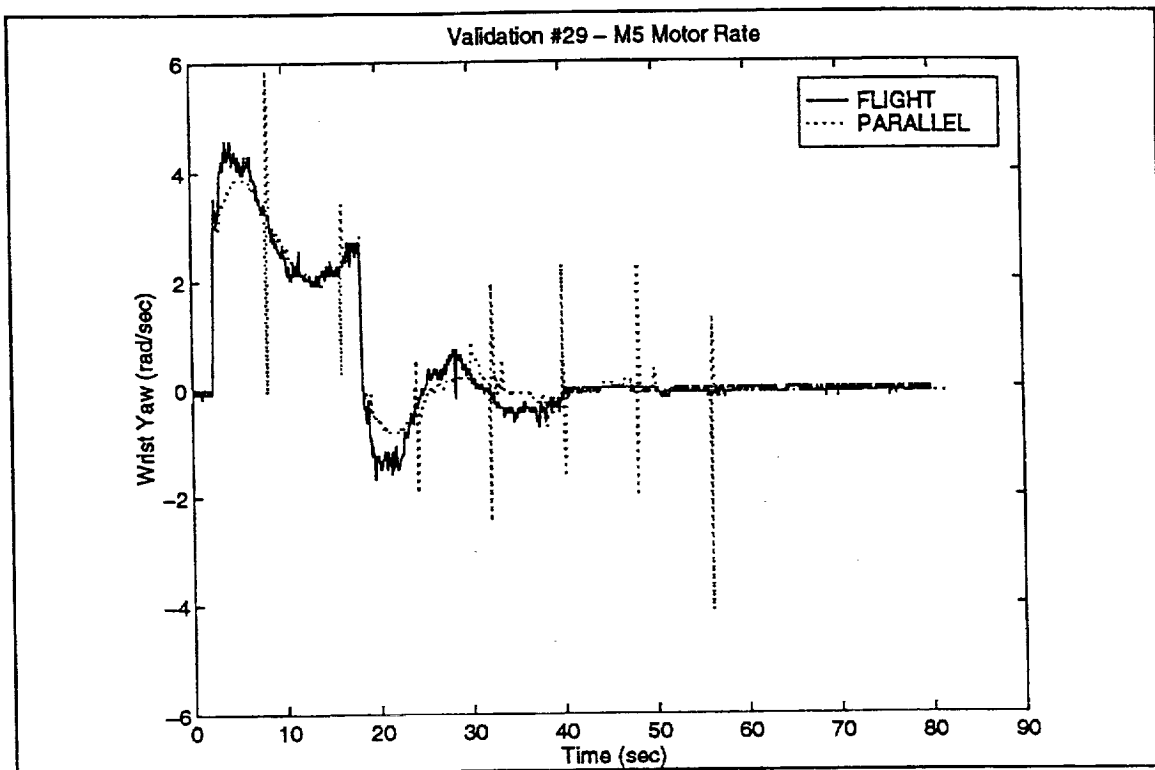


Figure D5: Validation #29 - M5 Motor Rate

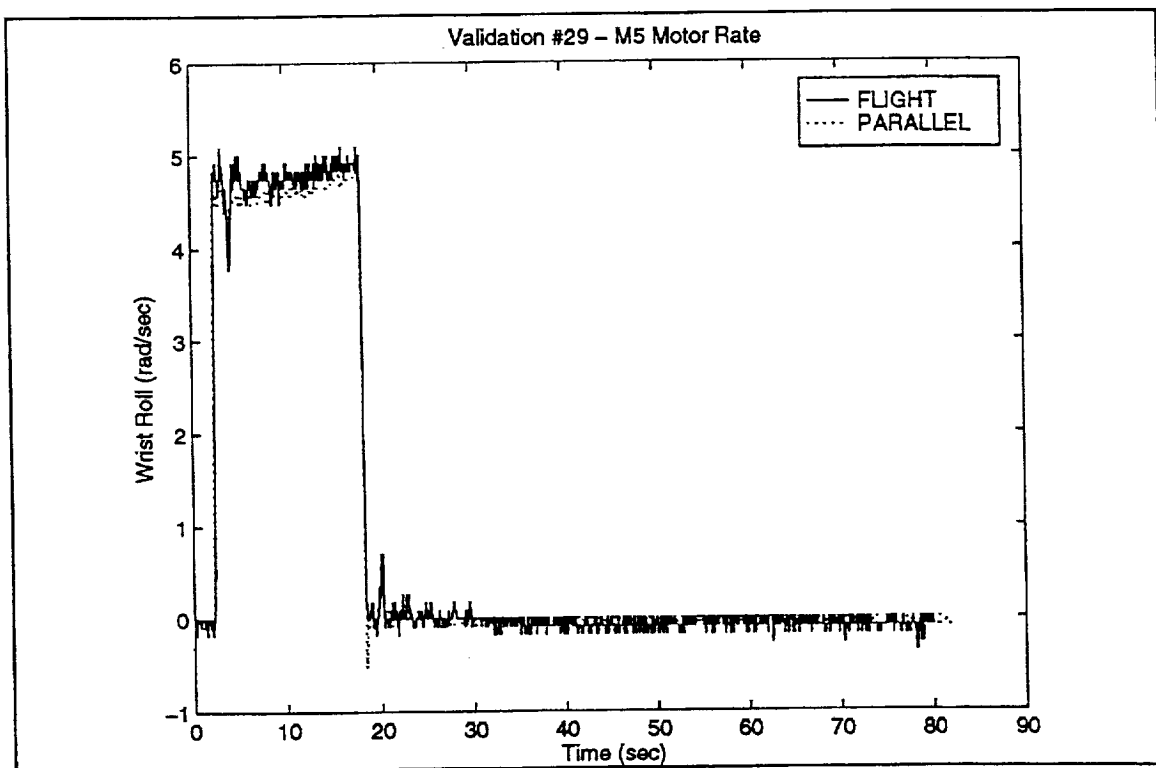


Figure D6: Validation #29 - M5 Motor Rate

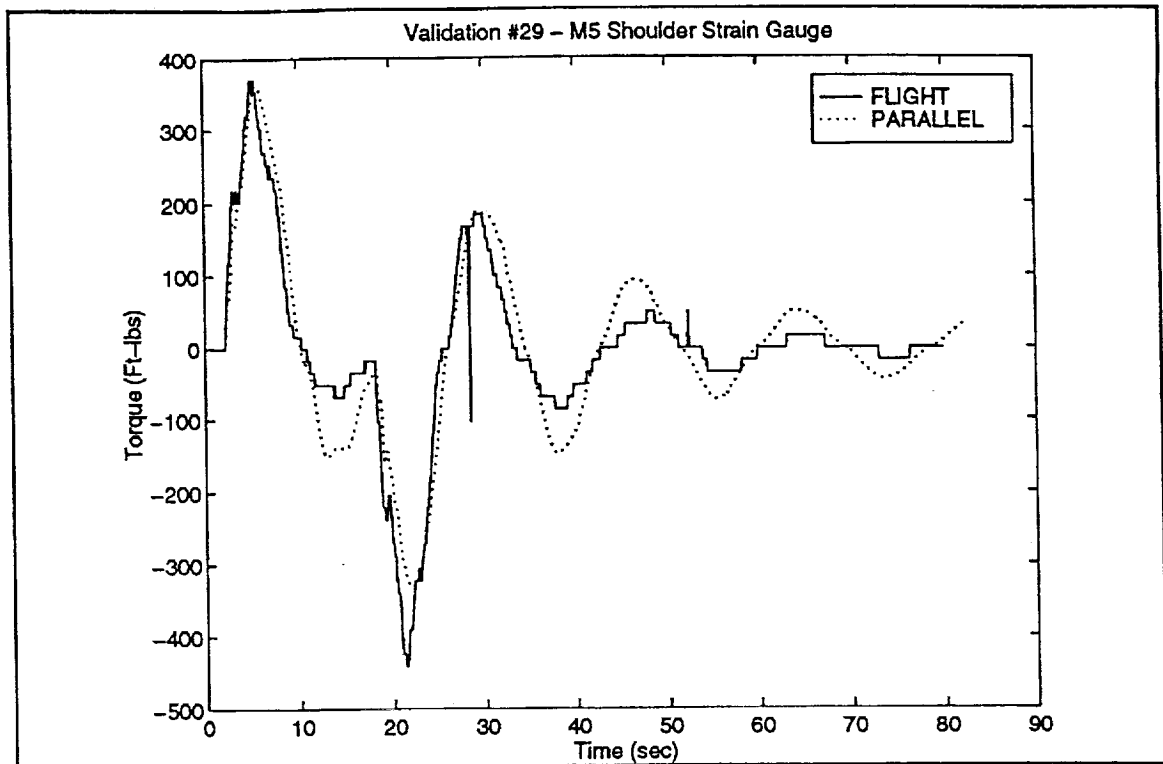


Figure D7: Validation #29 - M5 Shoulder Strain Gauge

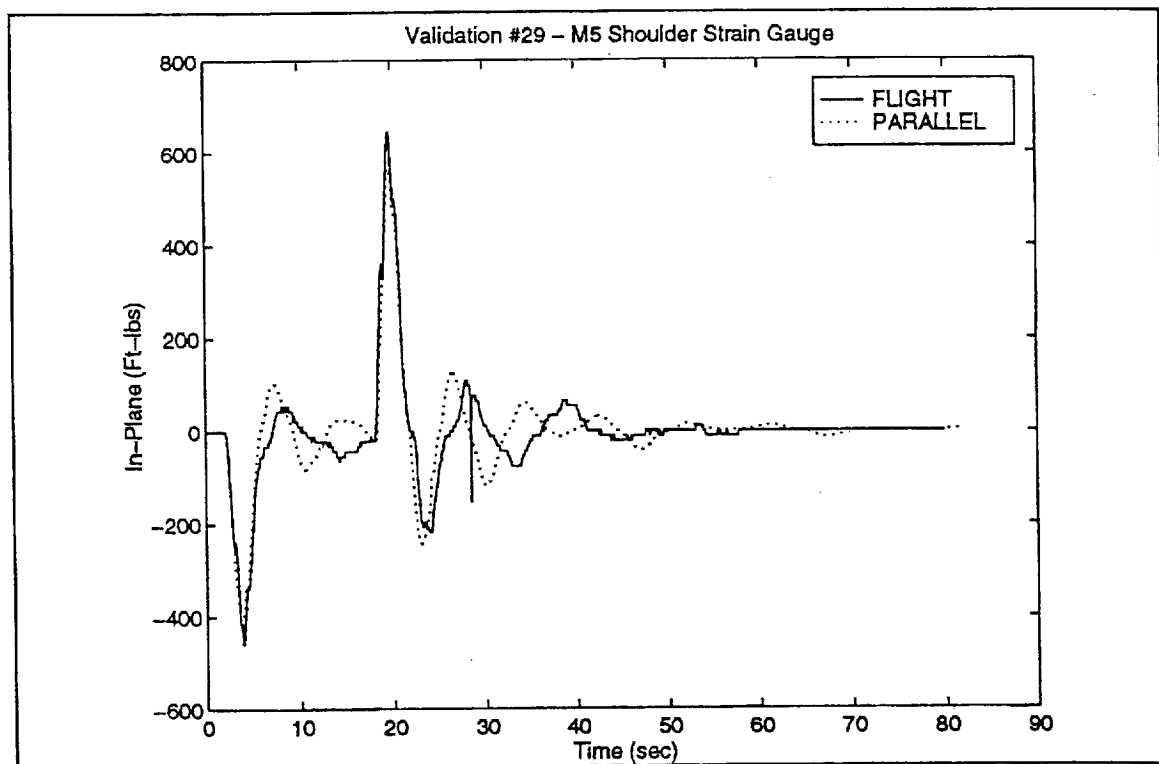


Figure D8: Validation #29 - M5 Shoulder Strain Gauge

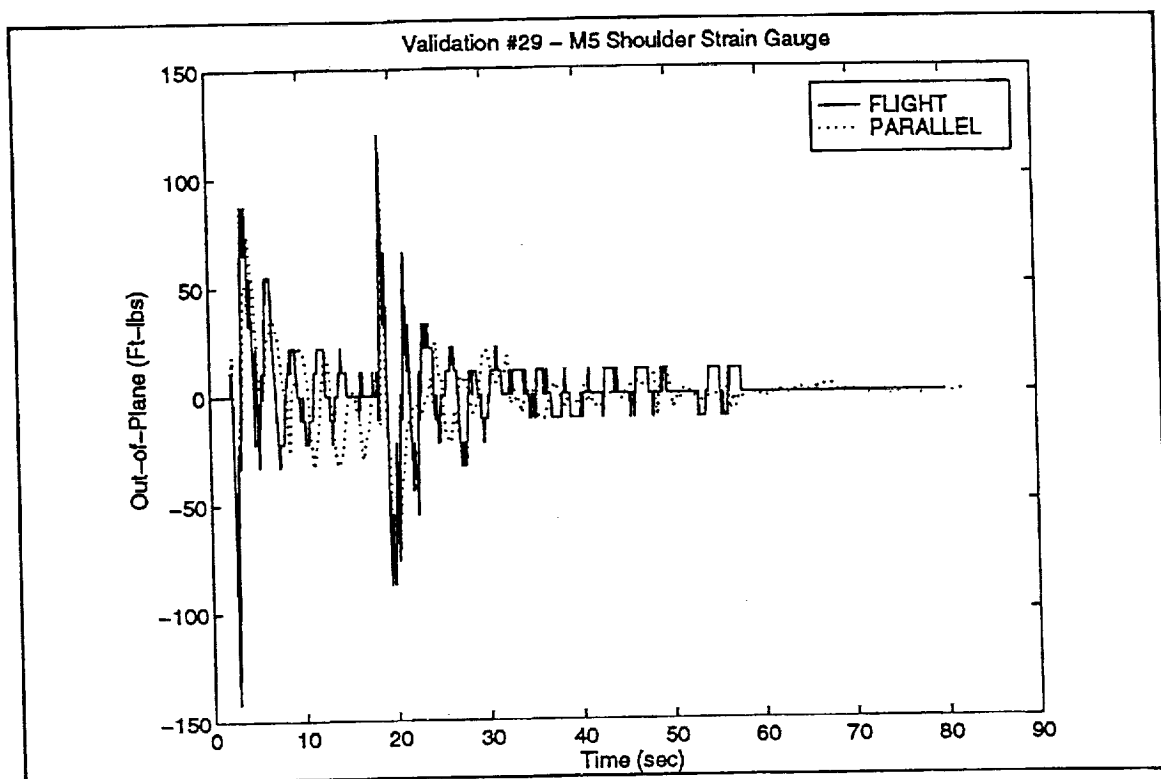


Figure D9: Validation #29 - M5 Shoulder Strain Gauge

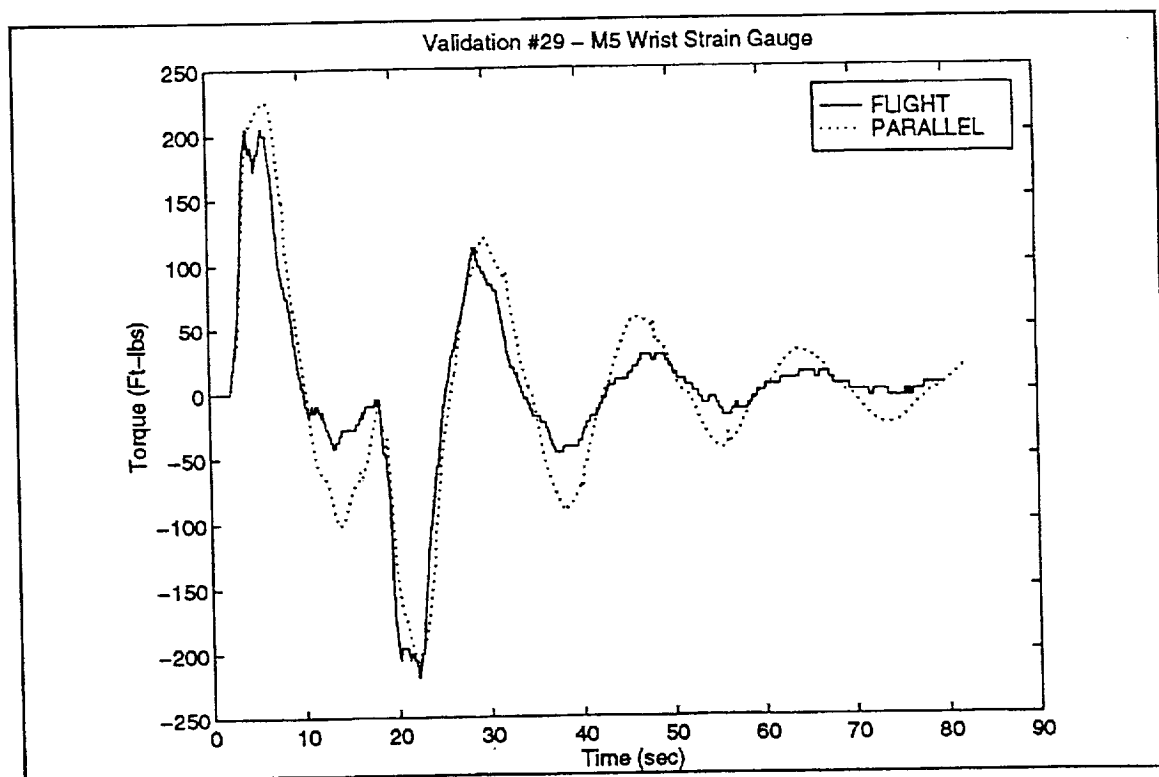


Figure D10: Validation #29 - M5 Wrist Strain Gauge

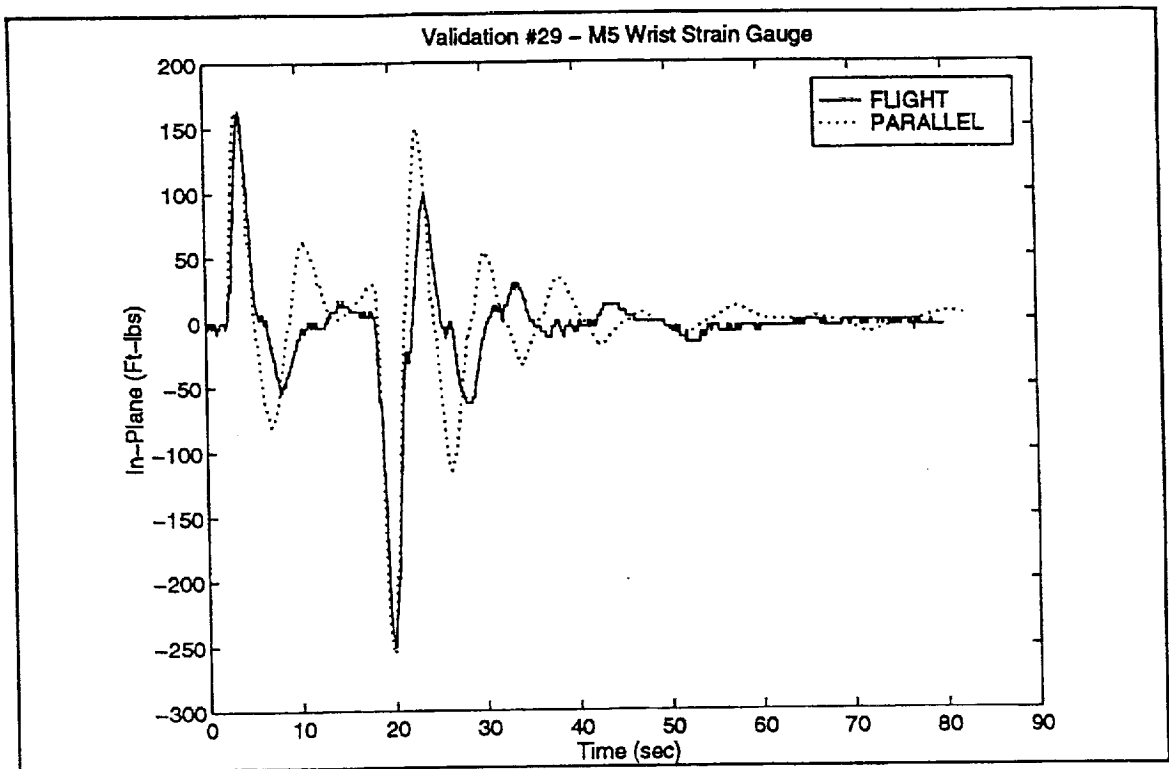


Figure D11: Validation #29 - M5 Wrist Strain Gauge

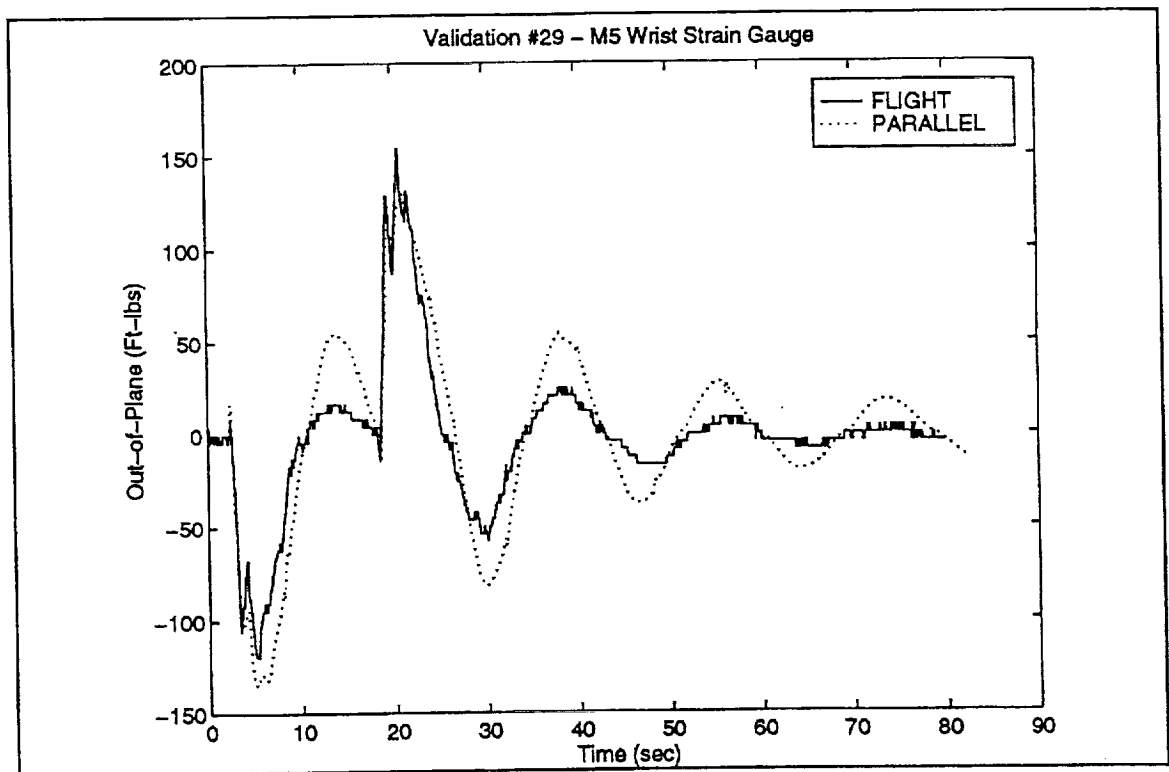


Figure D12: Validation #29 - M5 Wrist Strain Gauge

## Reference 4

DCI TM#081097  
DOTS RMS Script Commands  
August 10, 1997



To: bd Systems - Mr. Marlin Williamson  
From : Dynamic Concepts - Dr. Patrick Tobbe  
Subject : DOTS RMS Script Commands

## Introduction

In order to support flight testing of the Marshall Space Flight Center (MSFC) Video Guidance Sensor (VGS), two new commands were added to the Script mode of operation of the Dynamic Overhead Target Simulator (DOTS). These commands allow the user to position the DOTS through Remote Manipulator System (RMS) joint angles or Point of Resolution (POR) position and attitude data. This memorandum will briefly describe the software changes required for this capability and discuss the RMS script mode input data.

## RMS / DOTS Geometry

The RMS geometry is best described through the coordinate frames used by the kinematic flight software routines. Figure 1 illustrates the Orbiter Structural Reference Frame (OSR) and its position relative to the nose of the orbiter. Figure 2 depicts the Orbiter Body Axis System (OBAS) and Orbiter Rotation Axis System (ORAS). These frames are co-located with the OSR. Figure 3 shows the joint dynamics frames used to reference the RMS joint angles. The End Effector Operation (EEOP) and End Effector Reference (EERF) frames are co-located and pictured in Figure 4. The orientation of the co-located Payload Operating (PLOP) and Payload Reference (PLRF) frames of Figure 5 are fixed relative to each other, but are mission dependent with respect to the end effector frames.

Figure 6 integrates the RMS coordinate frames with the VGS and DOTS frames. The active side of the VGS, D2, and RMS OSR, OBAS, and ORAS frames are fixed relative to DOTS lab frame B0. D2 is assumed to be in the payload bay of the orbiter. The location and orientation of the passive VGS target, D1, and RMS end effector and payload frames are constant with respect to the DOTS end effector frame B8. D1 is assumed to be attached to the RMS payload as well as the DOTS mock up. The POR is also fixed relative to B8 and is not required to be co-located with D1.

The Script mode interface to the DOTS control software is the position vector of D1 with respect to D2,  $R_{D1\_D2}$ , and the orientation of D2 relative to D1,  $[D2\_D1]$ . The notation  $R_{A\_B\_C}$  will be used to describe the position of A with respect to B in C coordinates.  $[A\_B]$  is the transformation matrix from the B frame to the A frame.

The location of D1 with respect to D2 can be found from the vector equation

$$R_{D1\_D2} = R_{D1\_EEOP} - R_{POR\_EEOP} + R_{POR\_OSR} - R_{D2\_OSR}$$

where the C frame has been neglected from the notation. The orientation of D2 with respect to D1 is computed from

$$[D2\_D1] = [D2\_OSR][OSR\_ORAS][ORAS\_PLOP][PLOP\_D1]$$

## Software Modifications / Input Data

Four new files were added to the Script directory in order to incorporate this capability into the DOTS software. *RMS.INC* is the include file containing all variables for the RMS kinematics. *RMS\_INIT.F* is the initialization routine called by *SCRIPT.F* to read the namelist *RMSGEO* and set the constants used in the calculations. The namelist *RMSGEO* reads the file *RMS.INP* in the input directory. The following variables are initialized in *RMS.INP*.

$R_{D2\_OSR\_OSR}$  = Vector from OSR to D2 in OSR coordinates in feet

$[D2\_OSR]$  = Transformation matrix from OSR to D2

$R_{D1\_EEOP\_D1}$  = Vector from EEOP to D1 in D1 coordinates in feet

$[PLOP\_D1]$  = Transformation matrix from D1 to PLOP

$[PLOP\_EEOP]$  = Transformation matrix from EEOP to PLOP

$V_{EE\_POR}$  = Vector from end effector tip to POR in PLRF coordinates in feet

The file *SCRIPT.F* was modified to add the RMS script commands to the script parser. The RMS script tip command syntax is

*RMS T X Y Z  $\Theta_2$   $\Theta_3$   $\Theta_1$*

where *X Y Z* are the POR coordinates with respect to OBAS expressed in OBAS coordinates in inches.  $\Theta_2$   $\Theta_3$   $\Theta_1$  are the 2-3-1 Euler angles from ORAS to EEOP in degrees.

The RMS script joint command syntax is

*RMS J  $\Theta_1$   $\Theta_2$   $\Theta_3$   $\Theta_4$   $\Theta_5$   $\Theta_6$*

where  $\Theta_1$  through  $\Theta_6$  are the RMS joint angles in the joint dynamics frames in degrees.

The file *SCRIPT.F* was also modified to call the RMS kinematics routine *RMS\_KIN* while in RMS script mode. *SCRIPT.F* communicates with the DOTS control process, through the blackboards, to perform the RMS moves in joint space.

The RMS forward kinematics equations are in the file *KDG.F*. This subroutine computes the RMS POR position and orientation,  $R_{POR\_OBAS\_OBAS}$  and  $[PLOP\_ORAS]$ ,

from the joint angles of the script joint command. *RMS\_KIN.F* uses the POR position and attitude from the script tip command or the output of *KDG.F* to compute the relative position between D1 and D2.

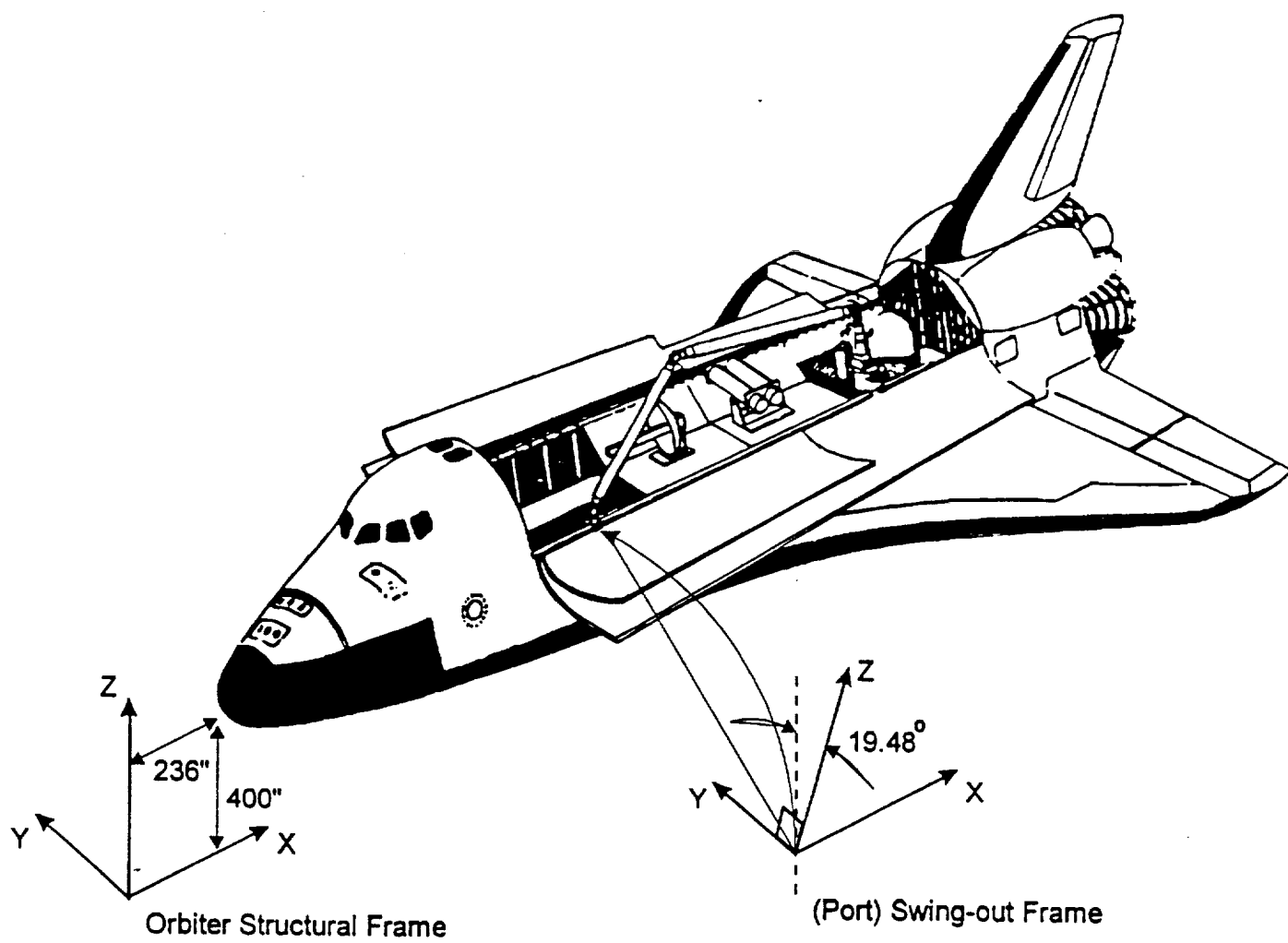


Figure 1

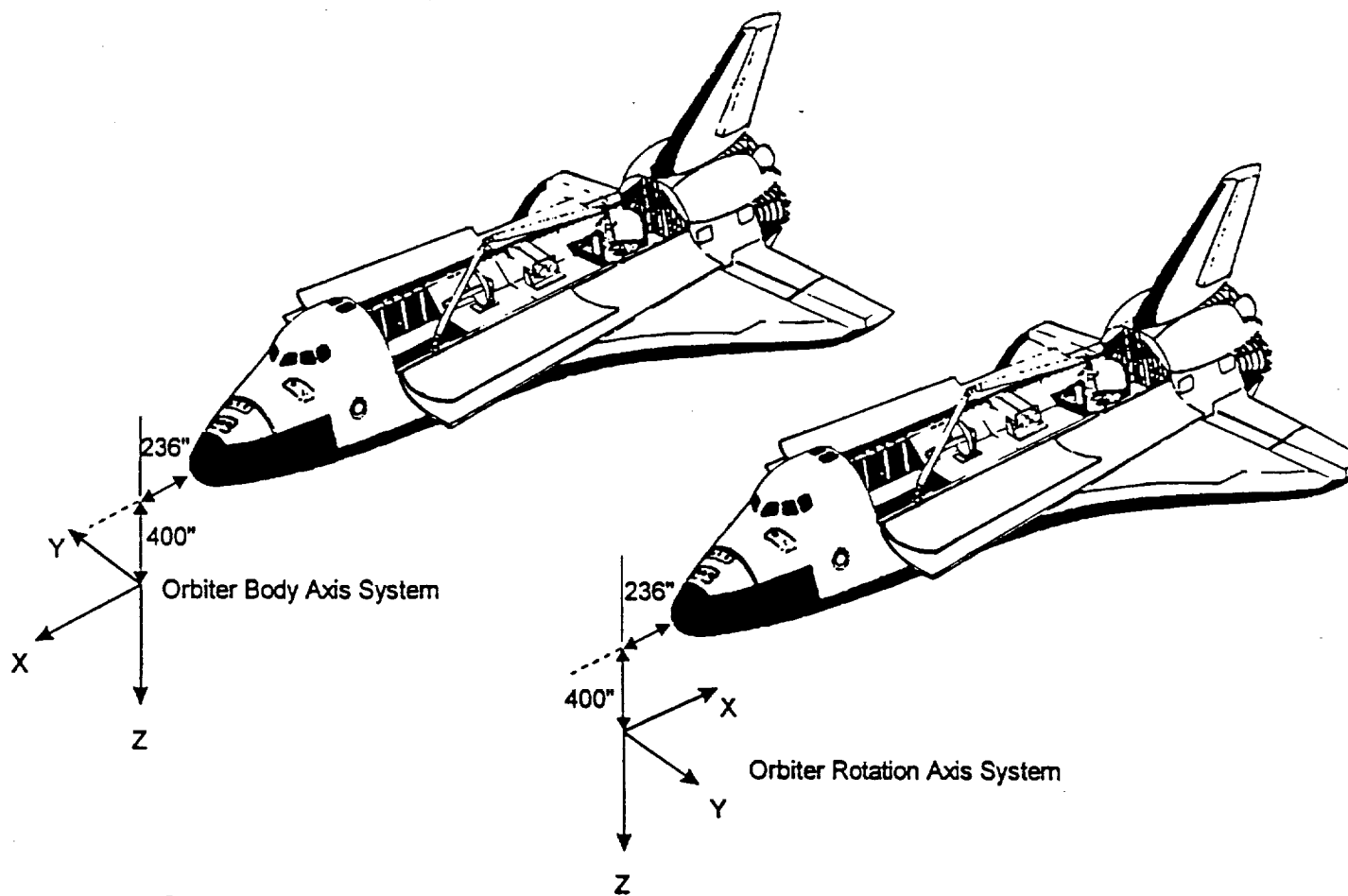
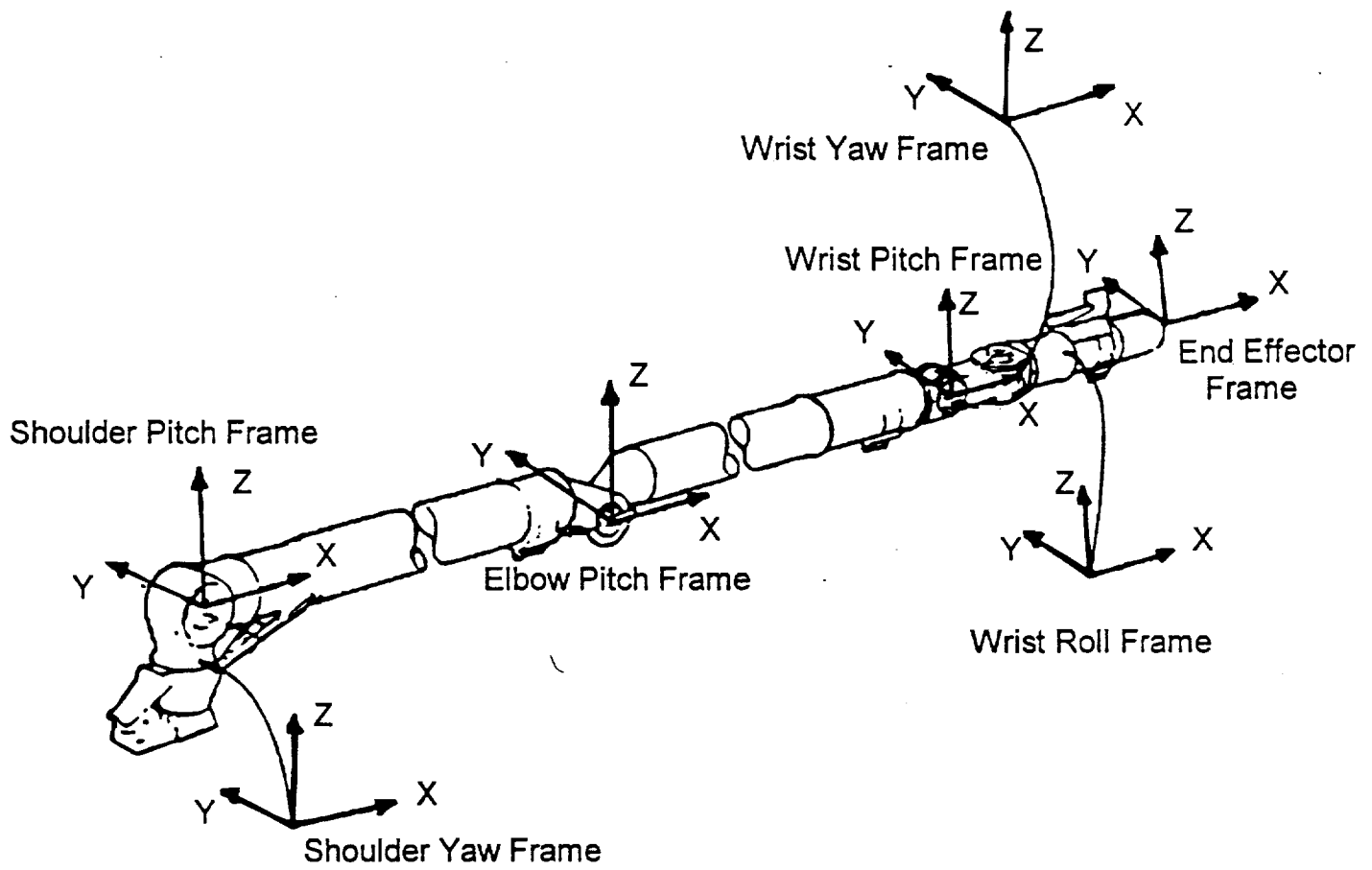
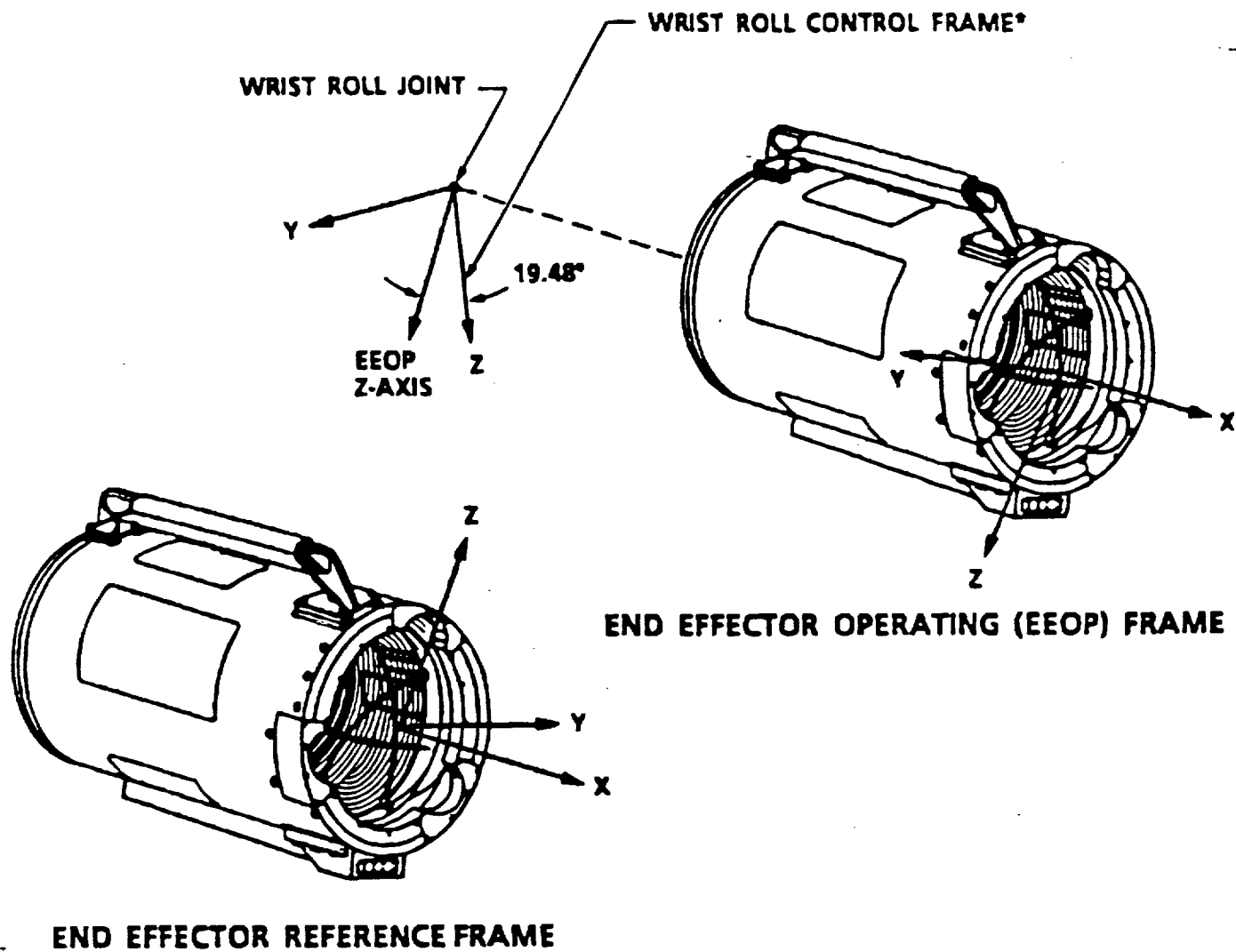


Figure 2



**SIX JOINT (DYNAMICS) FRAMES AND EE FRAME**

Figure 3



\*The wrist roll frame for the port arm is rolled 19.48° clockwise from the end effector frame (looking from the wrist roll joint to the end effector).

Figure 4

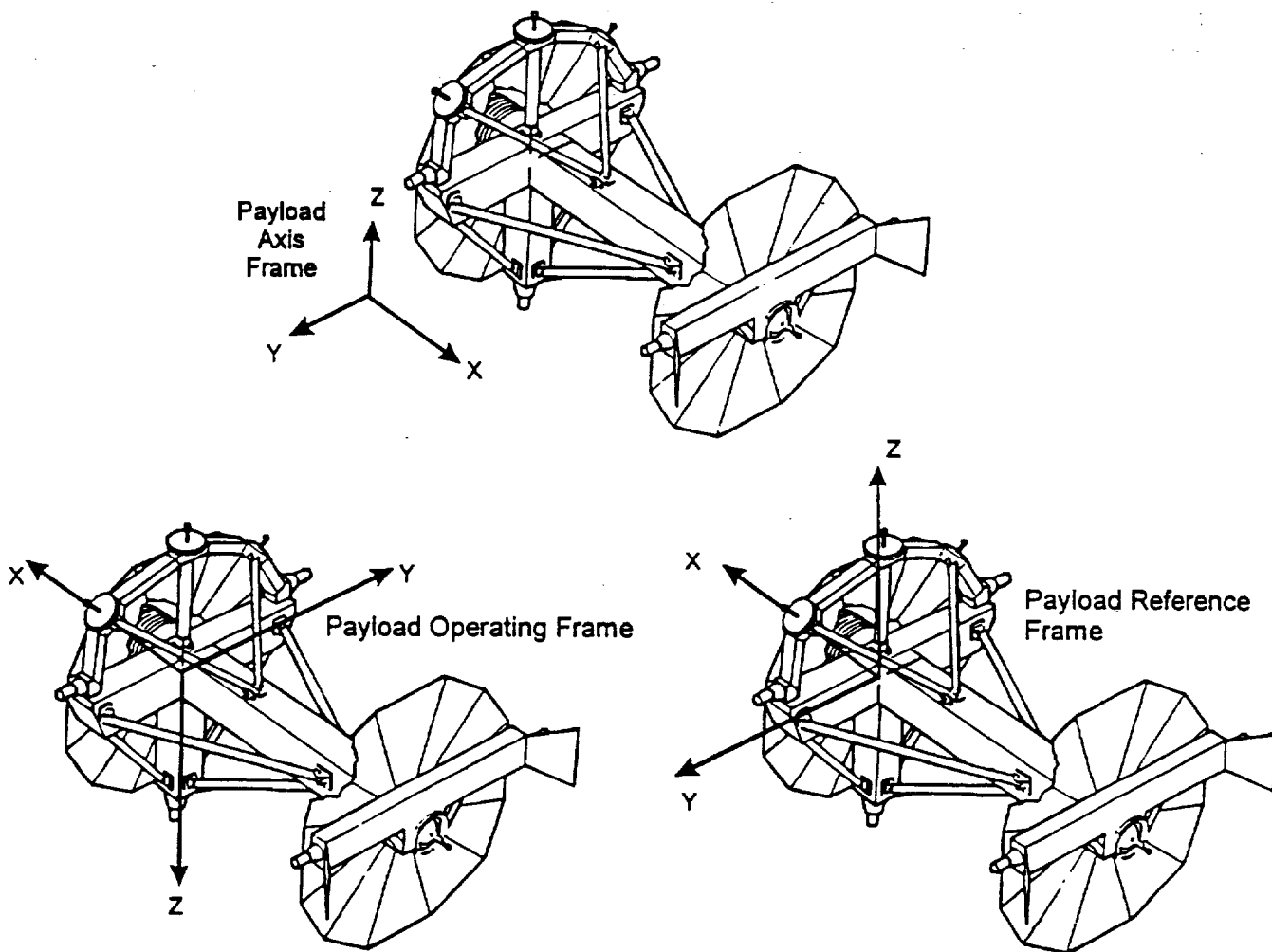


Figure 5



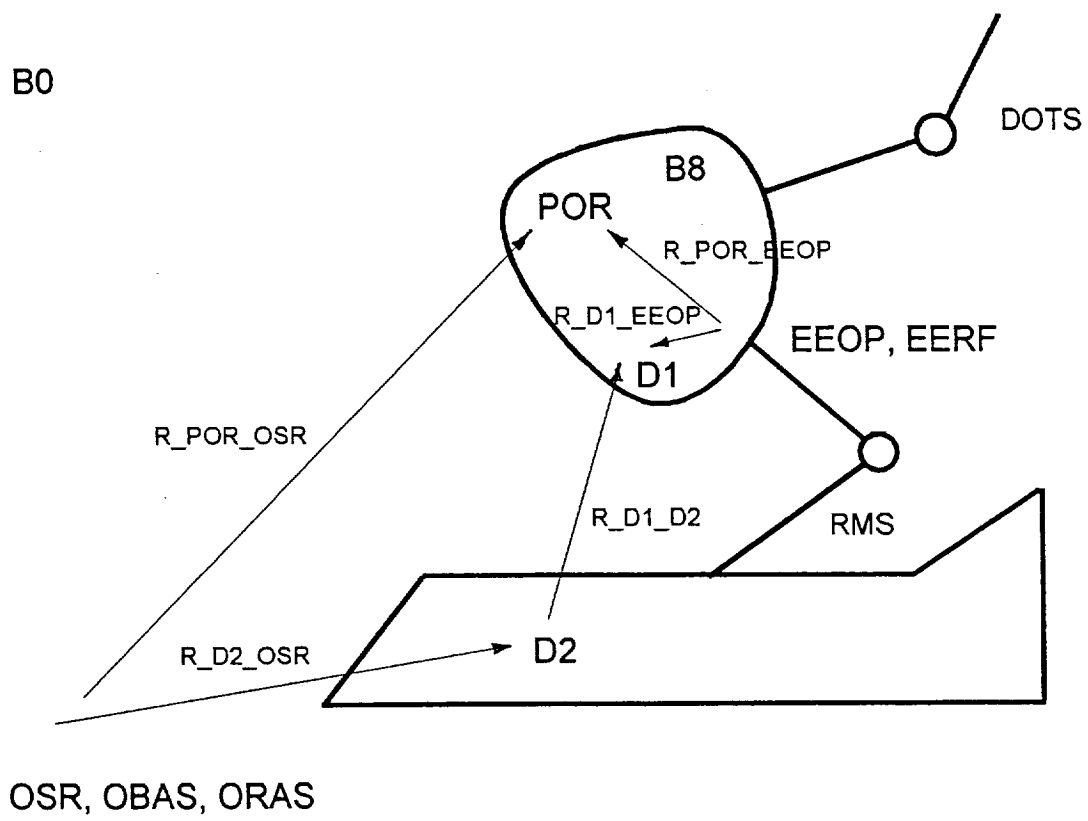


Figure 6

## Reference 5

DCI TM#092997  
ROCKET Flight-to-Sim Validation Runs  
September 29, 1997

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts - Dr. Patrick Tobbe  
Subject : *ROCKET* Flight to Simulation (FTS) Validation Runs

## Introduction

The real time Remote Manipulator System (RMS) simulation *ROCKET*, hosted on the SGI Challenge machine UQBAR, was modified to run a series of Flight to Simulation validation cases. These twenty-nine cases, shown in Table 1, are from the SPAR Payload Deployment and Retrieval System (PDRS) Simulation Database. These runs were previously made with the real time RMS simulation on the Alliant computer systems ALLI and CAT. This memorandum will describe software changes made to *ROCKET* to perform these runs and compare results to those from ALLI.

## Software Modifications

Upon completion of the initial software changes previously documented, *ROCKET* was modified to perform the FTS runs. This approach provides a software only verification of *ROCKET* after hosting on the SGI UQBAR.

Initial debugging consisted of modifying the path names in the make files and correcting syntax errors. Next, the C and FORTRAN structures used by the *ROCKET* C executive routine and the various FORTRAN processes were compared and corrected to describe the same variables. The subroutine *CALL* statements and subroutine argument lists were then verified to be the same. A temporary output routine was written for debugging and plotting purposes. The code was then successfully compiled and initial runs were made.

As expected, the FTS results did not initially agree between UQBAR and ALLI. It was eventually shown that each of the individual processes was operating correctly and the errors resulted from the process communication algorithms. To correct these problems, the following software changes were made. First, the *BE* and *GA* arrays were moved from the *RLN* to the *DtoR* structure for use in the *DYN* process. These variables are used by the *DYN* process to compute strain gage loads and RMS backdrive forces. It was then discovered that the *DtoC* structure was being toggled every 40 milliseconds instead of 80 milliseconds. This was fixed temporarily by setting the *CTL* process cycle time to 80 milliseconds in the routine *TCONFIG.C*. Bob Linner of MSFC is developing a more permanent solution for this problem in which the structure will be toggled every other cycle in the *CTL* process. The last communications problem was between the *RLN* and *DYN* processes. A flag was inserted into the *DtoR* structure and toggled by the two processes. This flag was used to initialize variables in the *DYN* process after a model update from *RLN*.

## Simulation Results

The FTS cases consist of twenty-nine simulation runs under the following four RMS modes of operation : brakes on, direct drive, single joint, manual augmented. All of these cases were run with *ROCKET* and compared with results from ALLI. Data was output from *ROCKET* every 2 milliseconds, while that from ALLI was written out every 80 milliseconds. Comparisons were made between strain gage loads, joint angles, commanded motor shaft rates, and tachometer outputs. All of the runs from *ROCKET* agreed with those made on ALLI and from the non-real time simulations. Figures 1 through 16 are the strain gage moments about Y and Z at the shoulder and wrist locations for runs 1, 7, 23, and 27. These figures show the correlation between data from ALLI and that from *ROCKET*. Run 1 is a brakes on, thruster firing case. Run 7 is a direct drive case, while run 23 is a single joint example. Run 27 is a man-in-the-loop manual augmented mode case.

In looking at these figures, in particular the shoulder torque about Y for case 1, an intermittent high frequency oscillation is seen in the *ROCKET* data. The outline of this anomaly is also seen in the ALLI data due to the output data rate. It has been determined that this effect is driven by the rigid body acceleration vectors of the RMS booms.

## Conclusions / Recommendations

The RMS real time simulation *ROCKET* has been successfully transferred from ALLI to UQBAR. *ROCKET* currently matches the output of the FTS cases generated on ALLI. The cycle time for the *DYN* process in a software only mode, with no data output, is 1.5 milliseconds. This is an improvement from the 4 millisecond time achieved on ALLI. Work should continue to reduce this time to allow a larger margin for hardware / software communications and minimize the numerical integration step size. Work should also continue to understand the high frequency oscillation present in the RMS boom bending moments. The current version of *ROCKET* should next be integrated with the new parallel output process and SIX DOF hardware communications routines. *ROCKET* can be transferred to the SGI Challenge machine TLON upon completing the interface to the output process.

**Table 1 - FLIGHT-TO-SIMULATION VALIDATION RUN MATRIX**

| Run | ID  | STS       | PL       | Command test/mode                                   | Termination condition | Command duration (seconds) | Run time (seconds) | Greenwich Mean Time (seconds) |
|-----|-----|-----------|----------|---|-----------------------|----------------------------|--------------------|-------------------------------|
| 1   | P2  | 4, case 2 | Unloaded | PRCS: -pitch (L3D, R3D, F3U)                        | Brakes                | 1.04 <sup>a</sup>          | 30                 | 15872909.660                  |
| 2   | P18 | 8         | PFTA2    | PRCS: +roll (L3D, R1U)                              | Brakes                | 0.24 <sup>a</sup>          | 145                | 21110790.192                  |
| 3   | P25 | 8         | PFTA5    | PRCS: -pitch (L3D, R3D)                             | Brakes                | 0.24 <sup>a</sup>          | 123                | 21210422.912                  |
| 4   | P1  | 4, case 2 | Unloaded | PRCS: +roll (L3D, R1U)                              | Brakes                | 1.04 <sup>a</sup>          | 32                 | 15872715.420                  |
| 5   | P20 | 8         | PFTA2    | PRCS: -pitch (L3D, R3D)                             | Brakes                | 0.24 <sup>a</sup>          | 117                | 21110488.352                  |
| 6   | P22 | 8         | PFTA5    | PRCS: -roll (L1U, R3D)                              | Brakes                | 0.24 <sup>a</sup>          | 121                | 21210738.192                  |
| 7   | D1  | 2         | Unloaded | Direct drive: +wrist pitch                          | Brakes                | 5.36                       | 30                 | 27446213.458                  |
| 8   | D2  | 2         | Unloaded | Direct drive: +shoulder pitch                       | Brakes                | 6.80                       | 30                 | 27445830.178                  |
| 9   | D4  | 4, case 1 | IECM     | Direct drive: +shoulder pitch                       | Brakes                | 20.08                      | 85                 | 15606165.180                  |
| 10  | D7  | 8         | PFTA2    | Direct drive: +shoulder pitch                       | Brakes                | 30.40                      | 145                | 21127354.853                  |
| 11  | D8  | 8         | PFTA5    | Direct drive: +shoulder pitch                       | Brakes                | 31.28                      | 130                | 21208895.333                  |
| 12  | P3  | 4, case 2 | Unloaded | PRCS: +roll (L3D, R1U)                              | Safing                | 1.04 <sup>a</sup>          | 33                 | 15873105.620                  |
| 13  | P28 | 8         | PFTA5    | PRCS: +roll (L3D, R1U)                              | Safing                | 0.24 <sup>a</sup>          | 136                | 21211502.032                  |
| 14  | P29 | 8         | PFTA5    | PRCS: +pitch (L1U, R1U)                             | Safing                | 0.24 <sup>a</sup>          | 137                | 21211161.272                  |
| 15  | P23 | 8         | PFTA5    | PRCS: +roll (single drive <sup>b</sup> / L3D, R1U)  | Position hold         | 0.24 <sup>a</sup>          | 102                | 21212026.432                  |
| 16  | P26 | 8         | PFTA5    | PRCS: -pitch (single drive <sup>b</sup> / L3D, R3D) | Position hold         | 0.24 <sup>a</sup>          | 115                | 21211759.952                  |
| 17  | S2  | 7         | SPAS     | Single drive <sup>b</sup> : +shoulder pitch         | Safing                | 15.68                      | 65                 | 15067777.153                  |
| 18  | S5  | 8         | PFTA2    | Single drive <sup>b</sup> : +shoulder pitch         | Safing                | 15.44                      | 90                 | 21127160.613                  |
| 19  | S7  | 8         | PFTA2    | Single drive <sup>b</sup> : +wrist yaw              | Brakes                | 15.76                      | 59                 | 21125449.613                  |
| 20  | S8  | 8         | PFTA2    | Single drive <sup>b</sup> : +wrist roll             | Brakes                | 15.56                      | 76                 | 21125674.773                  |

<sup>a</sup>Jet firing termination

<sup>b</sup>Coarse rate.

Table 1 - FLIGHT-TO-SIMULATION VALIDATION RUN MATRIX (CONT'D)

| Run | ID  | STS | PL       | Command test/mode   | Termination condition | Command duration (seconds)    | Run time (seconds) | Greenwich Mean Time (seconds) |
|-----|-----|-----|----------|---|-----------------------|-------------------------------|--------------------|-------------------------------|
| 21  | S9  | 8   | PFTA5    | Single drive <sup>b</sup> :<br>+shoulder pitch                | Safing                | 18.24                         | 96                 | 21208656.533                  |
| 22  | S11 | 8   | PFTA5    | Single drive <sup>b</sup> :<br>+wrist yaw                     | Brakes                | 15.12                         | 74                 | 21206768.573                  |
| 23  | S13 | 2   | Unloaded | Single drive <sup>b</sup> :<br>+shoulder pitch                | Position hold         | 11.44                         | 45                 | 27446452.138                  |
| 24  | S14 | 2   | Unloaded | Single drive:<br>+elbow pitch                                 | Position hold         | 0.80 vernier<br>7.92 coarse   | 60                 | 27446560.418                  |
| 25  | S15 | 2   | Unloaded | Single drive <sup>b</sup> :<br>+wrist pitch                   | Position hold         | 9.68                          | 60                 | 27446759.218                  |
| 26  | A27 | 8   | PFTA5    | RHC <sup>b</sup> :+yaw,<br>ramp to +107<br>counts at 0.96 sec | Brakes                | 16.24 command<br>16.36 brakes | 108                | 21206339.373                  |
| 27  | M3  | 8   | PFTA5    | THC <sup>b</sup> :+Z,<br>ramp to -103<br>counts at 0.40 sec   | Brakes                | 15.72 command<br>15.84 brakes | 78                 | 21205737.333                  |
| 28  | M4  | 8   | PFTA2    | THC <sup>b</sup> :+Y,<br>ramp to +105<br>counts at 0.40 sec   | Position hold         | 15.40                         | 85                 | 21124426.933                  |
| 29  | M5  | 8   | PFTA5    | THC <sup>b</sup> :+Y,<br>ramp to +105<br>counts at 0.40 sec   | Position hold         | 16.00                         | 82                 | 21205945.933                  |

<sup>a</sup>Jet firing termination

<sup>b</sup>Coarse rate.

SIMSGI-Wrist Y Moment

val01  
1 ○ — 33  
val01nac  
1 △ - - - - 33

1 □ ..... 25

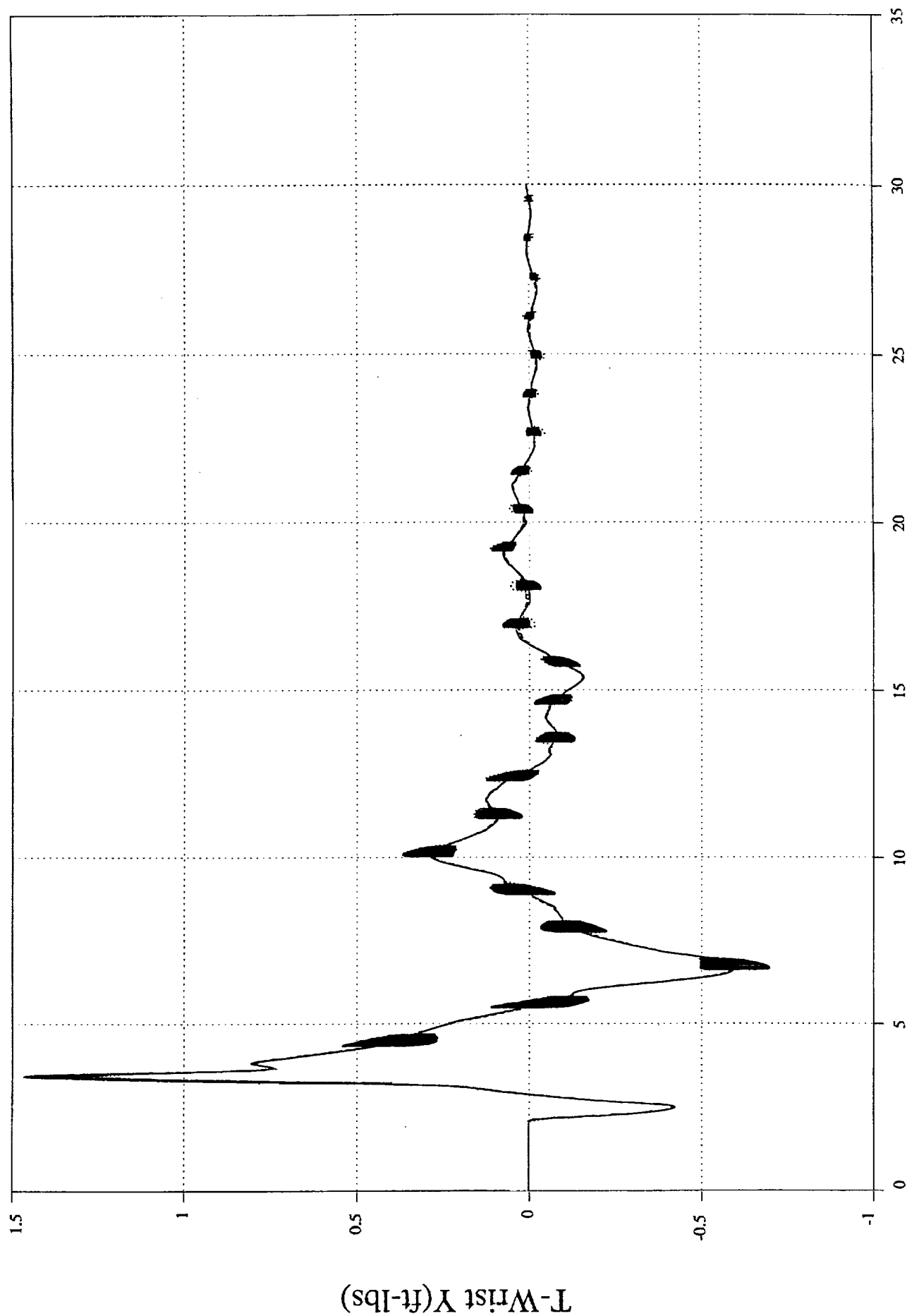


Figure 1  
TIME(sec)--->case01

SIMSGI-Wrist Z Moment

val01 1 34  
val01nac 1 34

1 26

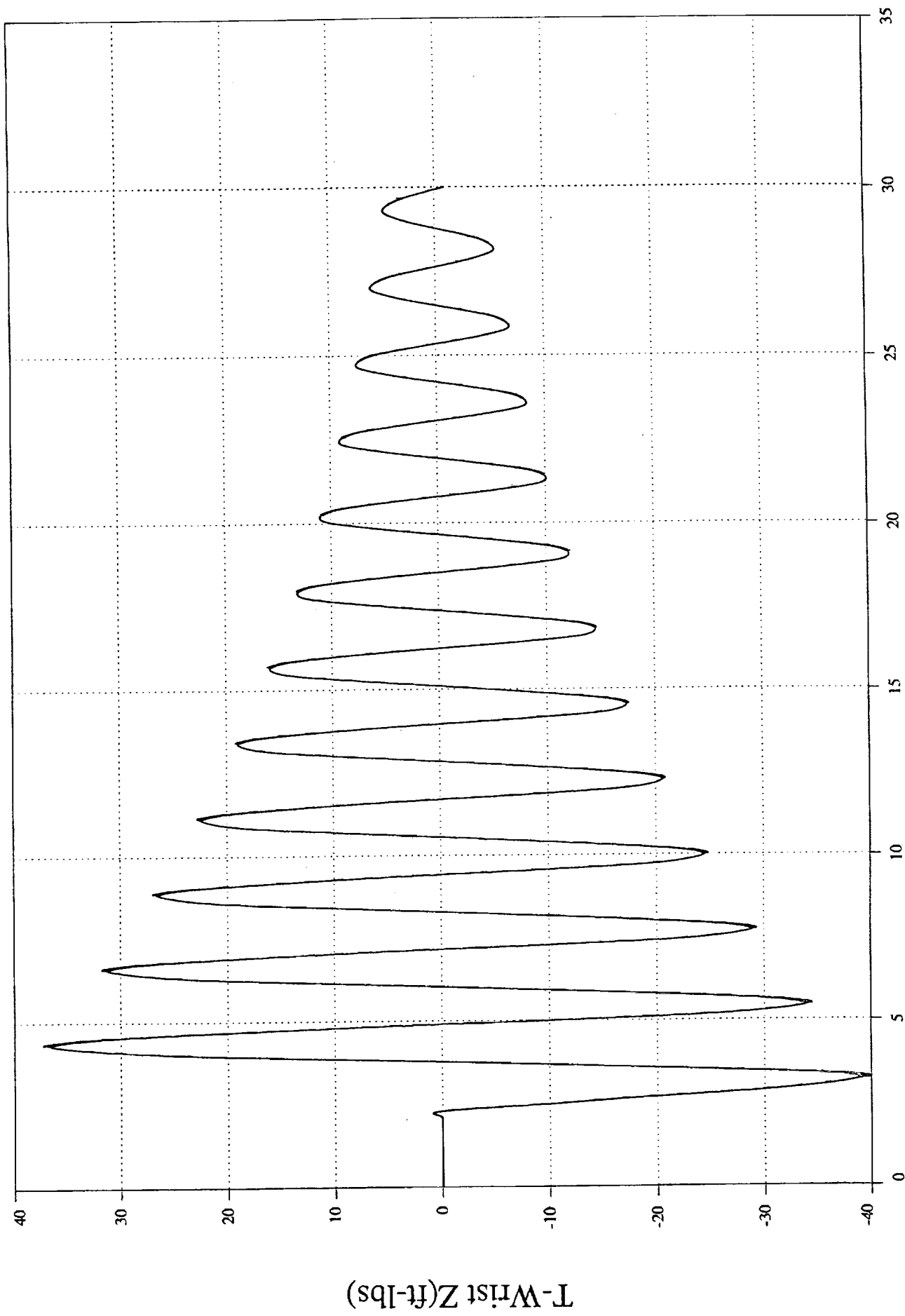


Figure 2

TIME(sec)--->case01



val01 1 0 39 val01nac 1 Δ 39 SIMSGI-Shoulder Y Moment 1 □ 19

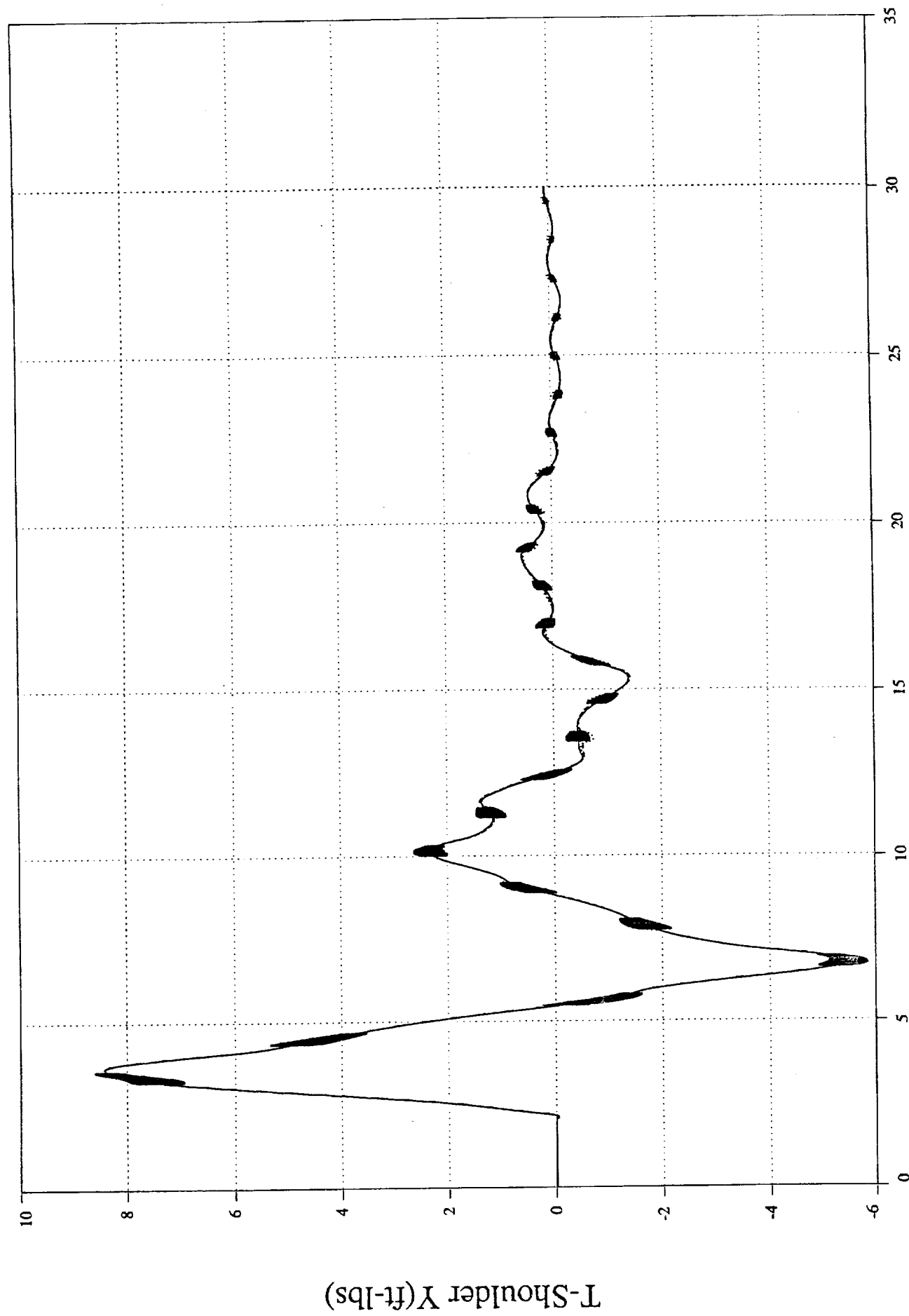


Figure 3

TIME(sec)--->case01



SIMSGI-Wrist Y Moment

val08 1 33  
val08nac 1 33

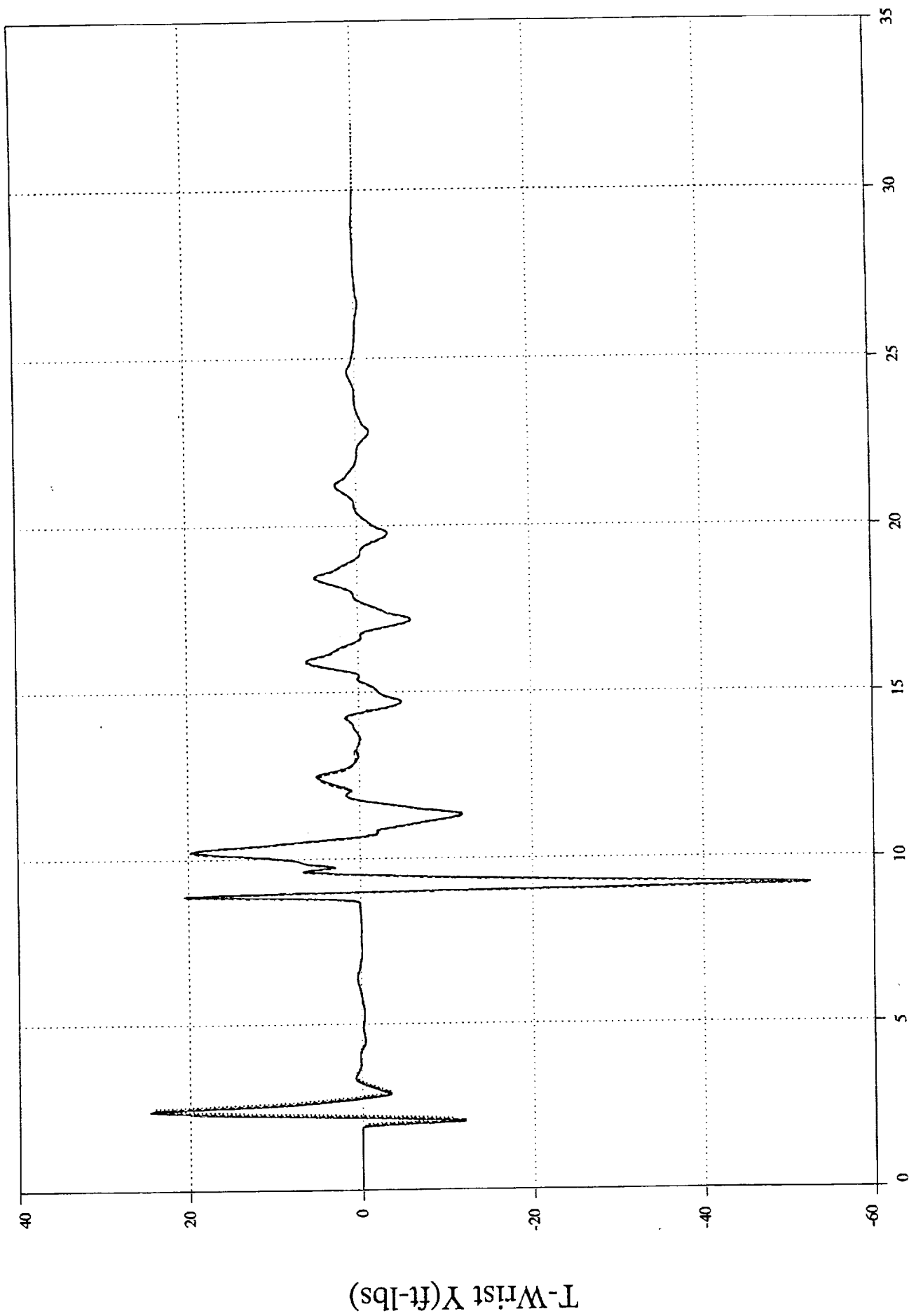


Figure 5

SIMSGI-Wrist Z Moment t/././fsval/v8hc

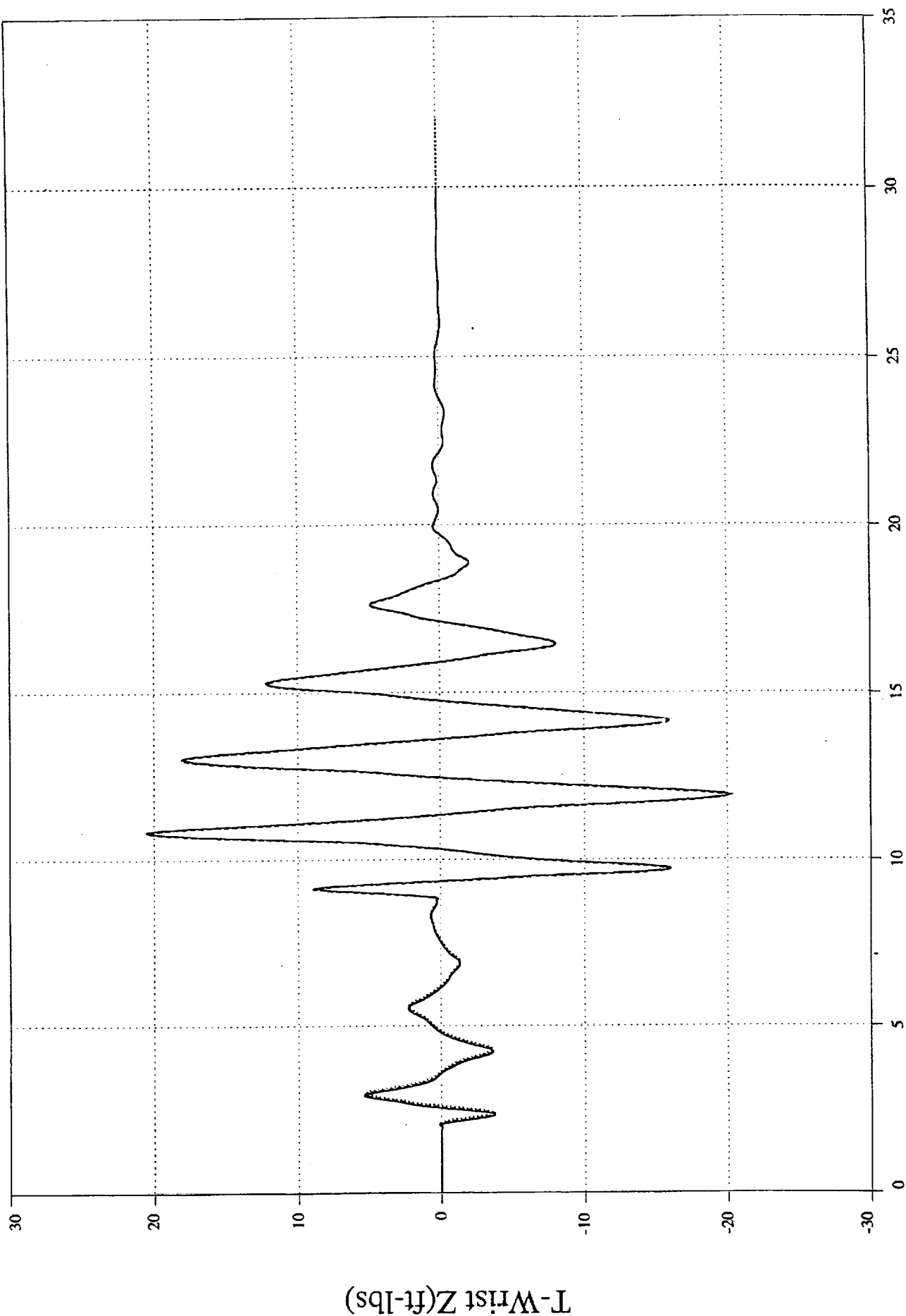
val08nac

val08

1 □ ..... □ 26

1 Δ-----Δ 34

1 ○-----○ 34



TIME(sec)--->case08

Figure 6

val08 1 ○ — ○ 39

val08nac 1 Δ — Δ 39

SIMSGI-Shoulder Y Moment #fsva/v8hc 1 □ — □ 19

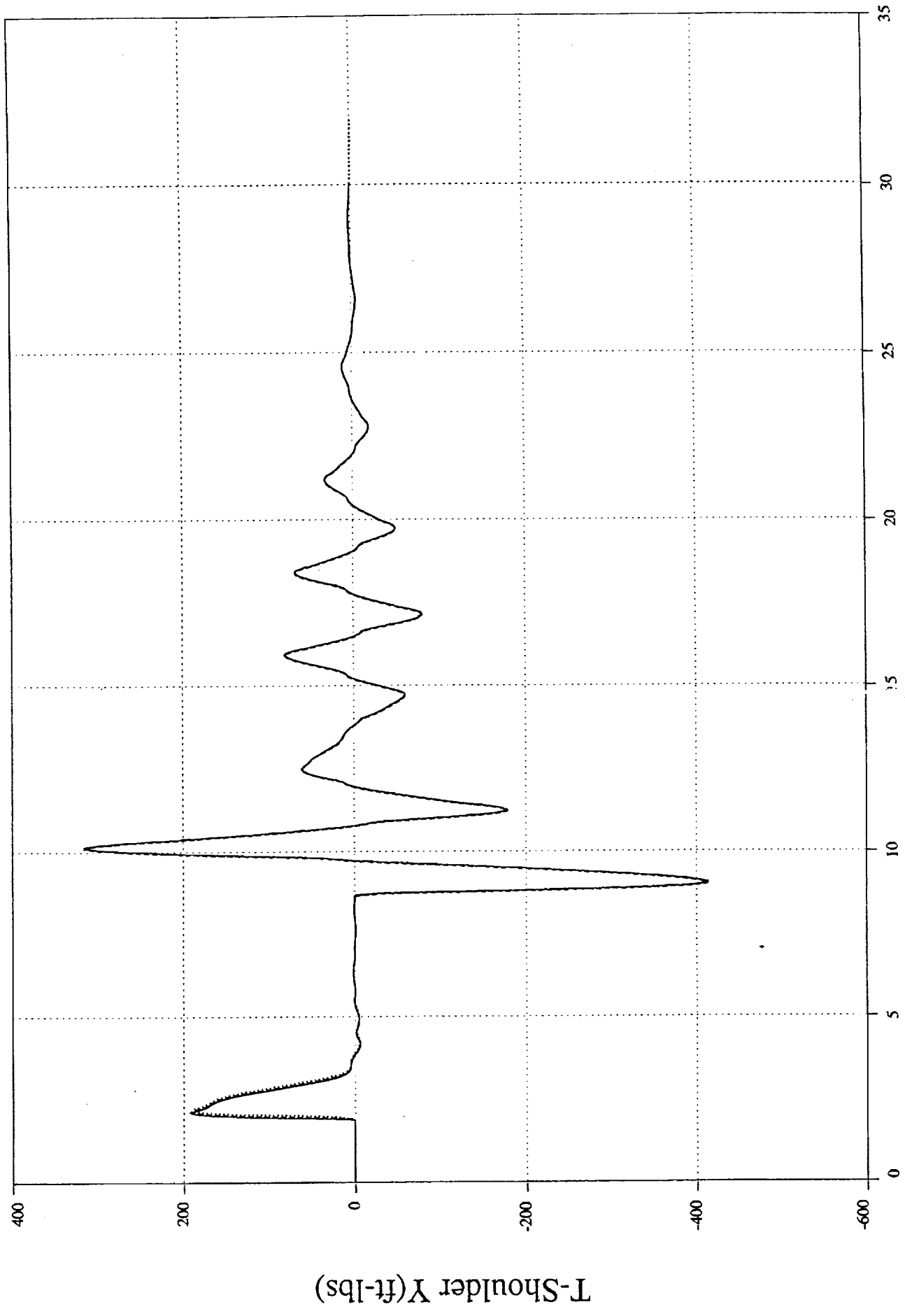


Figure 7  
TIME(sec)--->case08

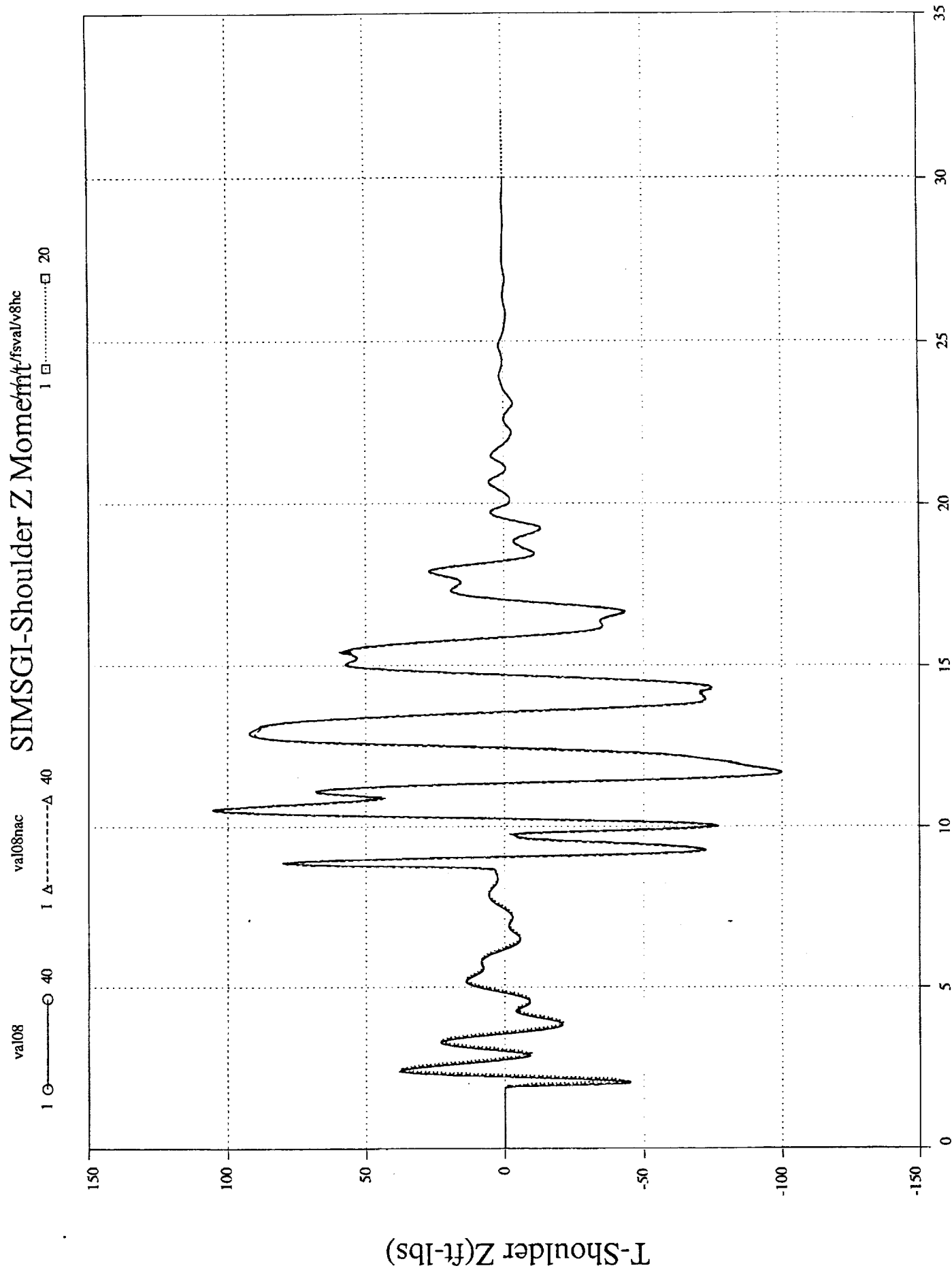


Figure 8

val23 1 33 33 val23nac 1 33 33 SIMSGI-Wrist Y Moment v23hc 1 25 25

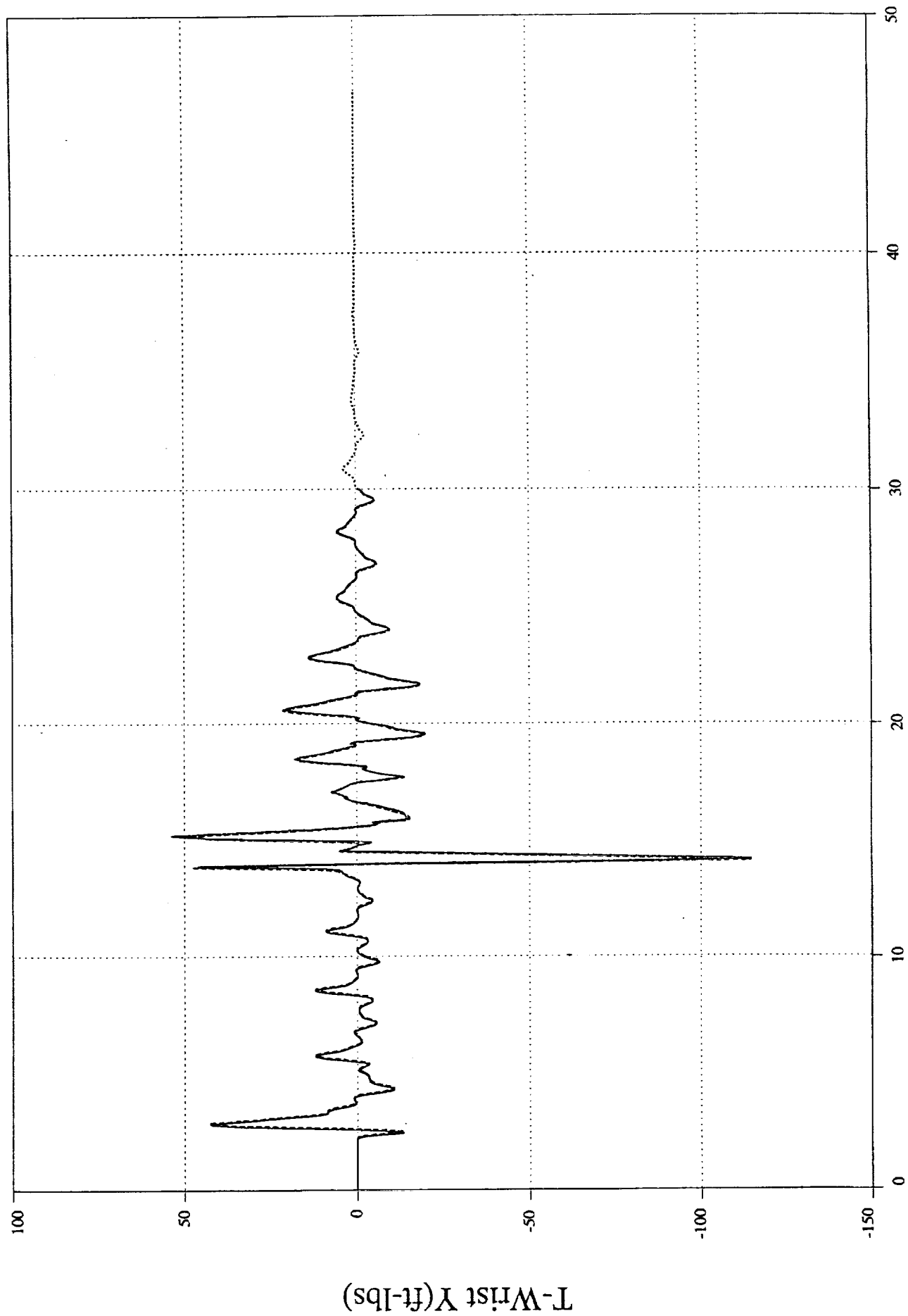


Figure 9

TIME(sec)--->case23

val23 1 0 34 val23nac 1 Δ 34 SIMSGI-Wrist Z Moment v23hc 1 □ 26

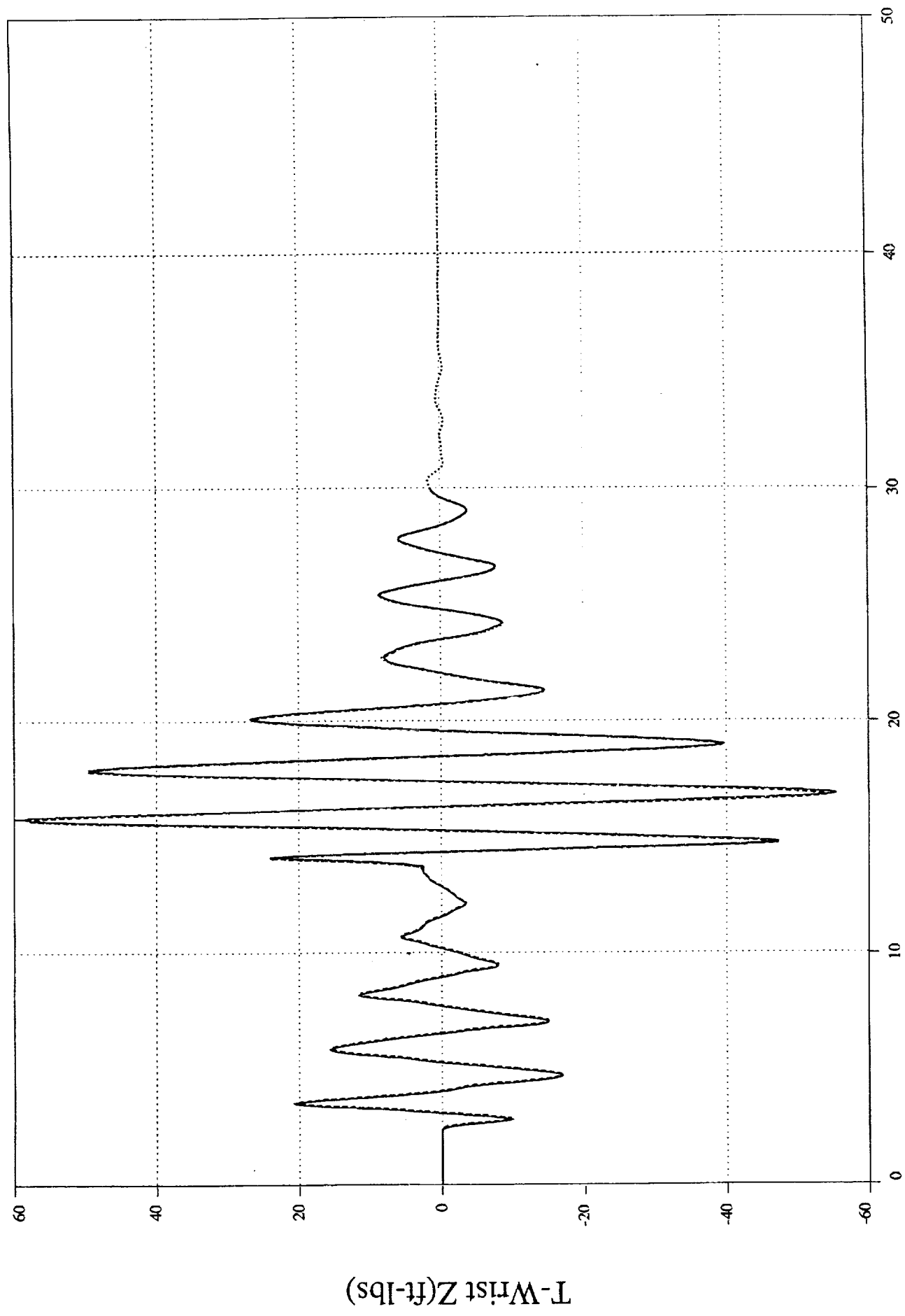


Figure 10



val23                      val23nac                      SIMSGI-Shoulder Y Moment                      v23hc  
1 ○-----○ 39                      1 Δ-----Δ 39                      1 □-----□ 19

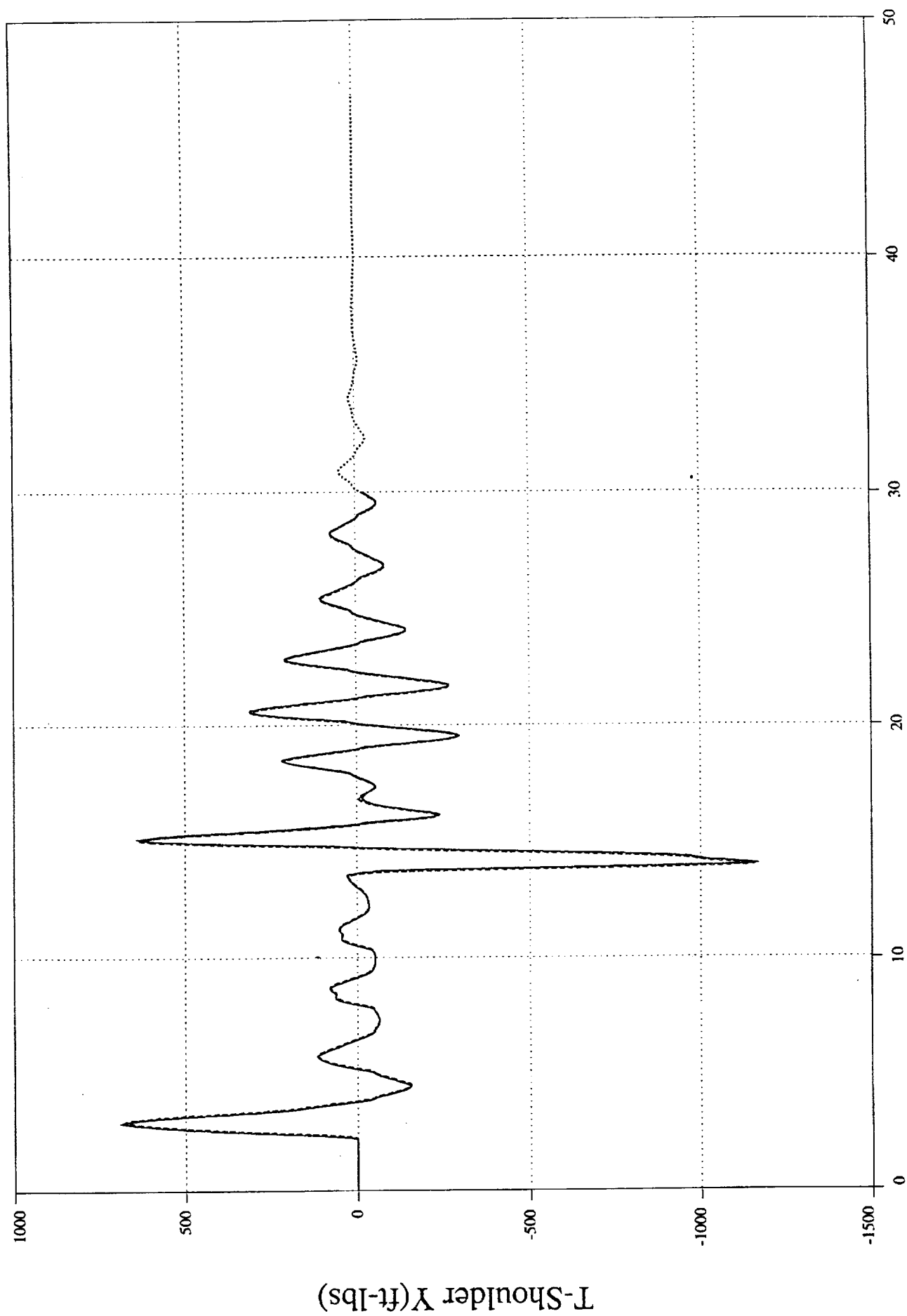


Figure 11

val23 1 G 40  
val23nac 1 Δ 40  
SIMSGI-Shoulder Z Momemt v23hc 1 □ 20

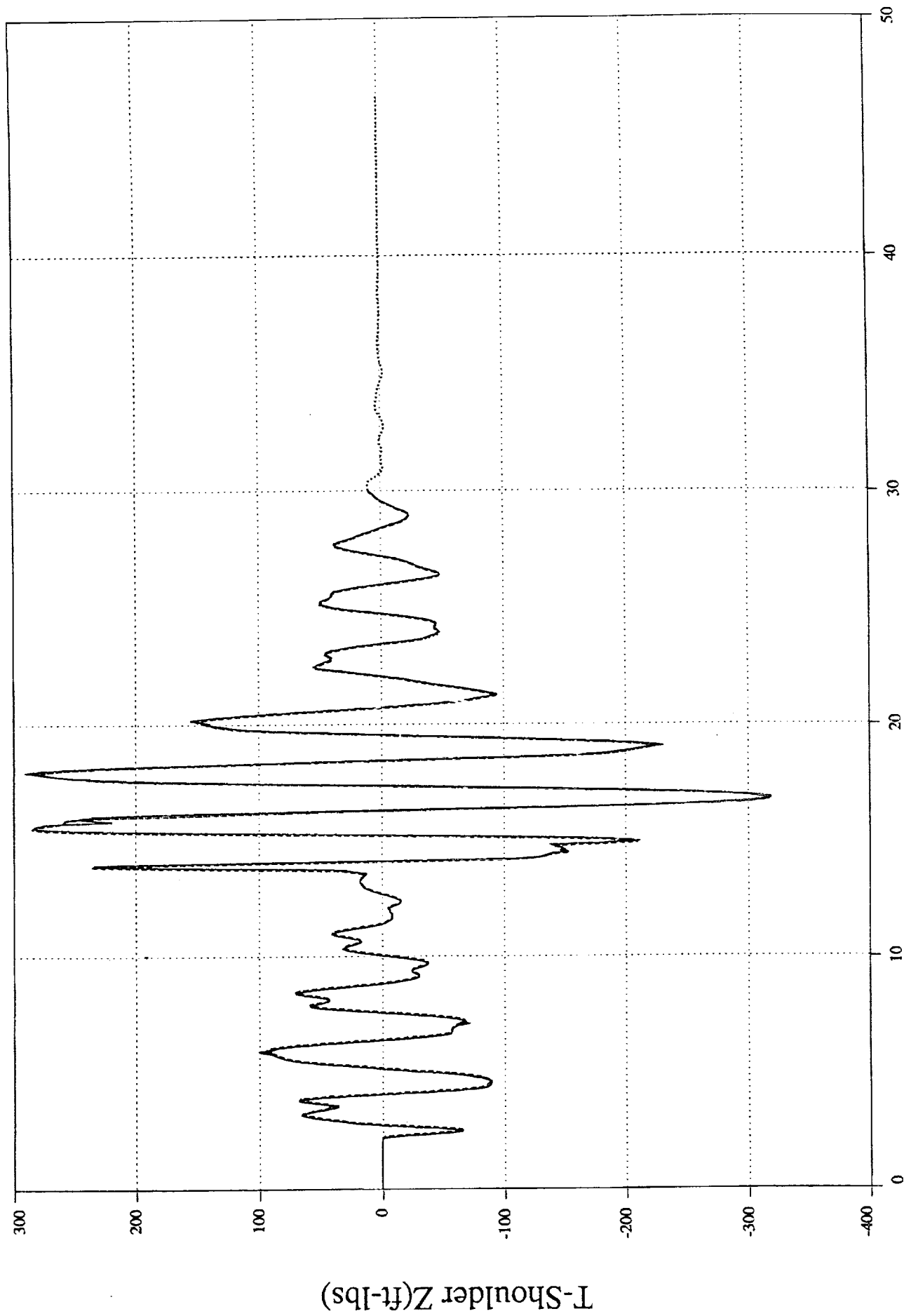


Figure 12

TIME(sec)--->case23

val27  $\bigcirc$  33  
 SIMSGI-Wrist Y Moment  $\bigcirc$  1  $\square$  25  
 val27nac  $\Delta$  33

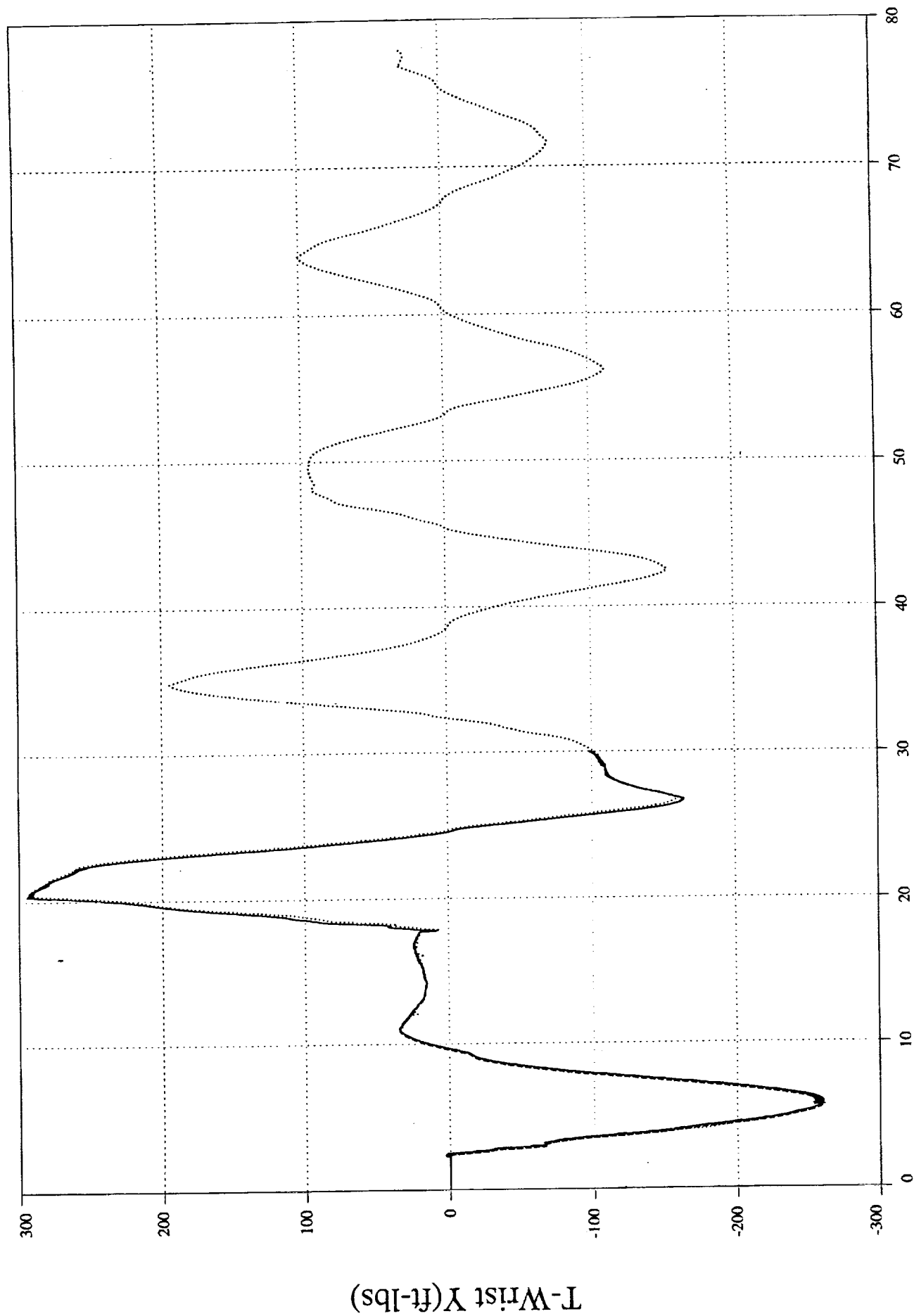


Figure 13

TIME(sec)--->case27

val27 1 ○ — ○ 34  
val27nac 1 Δ — Δ 34  
SIMSGI-Wrist Z Moment 1 □ — □ 26 v27hc

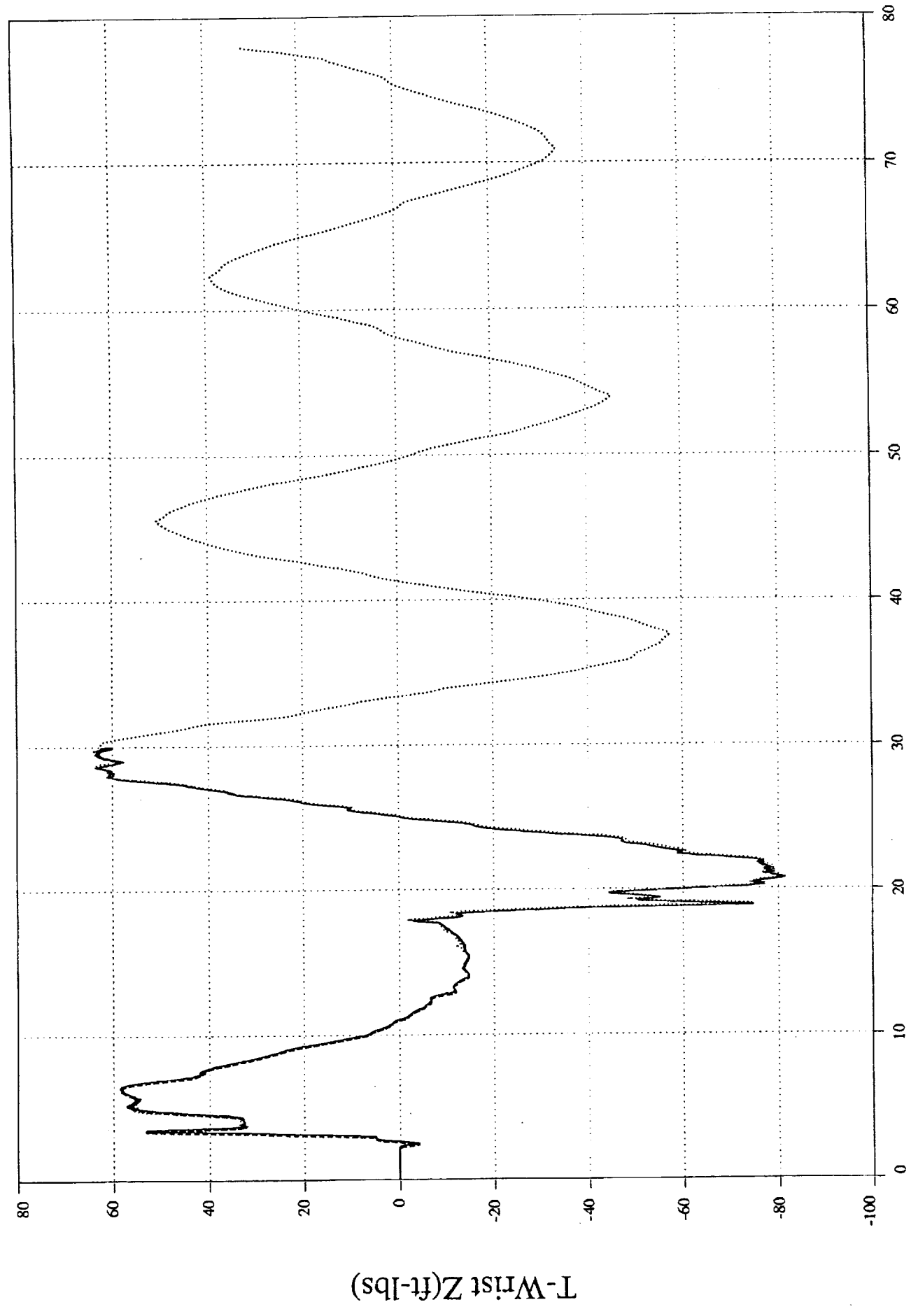


Figure 14  
TIME(sec)--->case27

val27 1 ○ 39  
val27nac 1 Δ 39  
SIMSGI-Shoulder Y Moment v27hc 1 □ 19

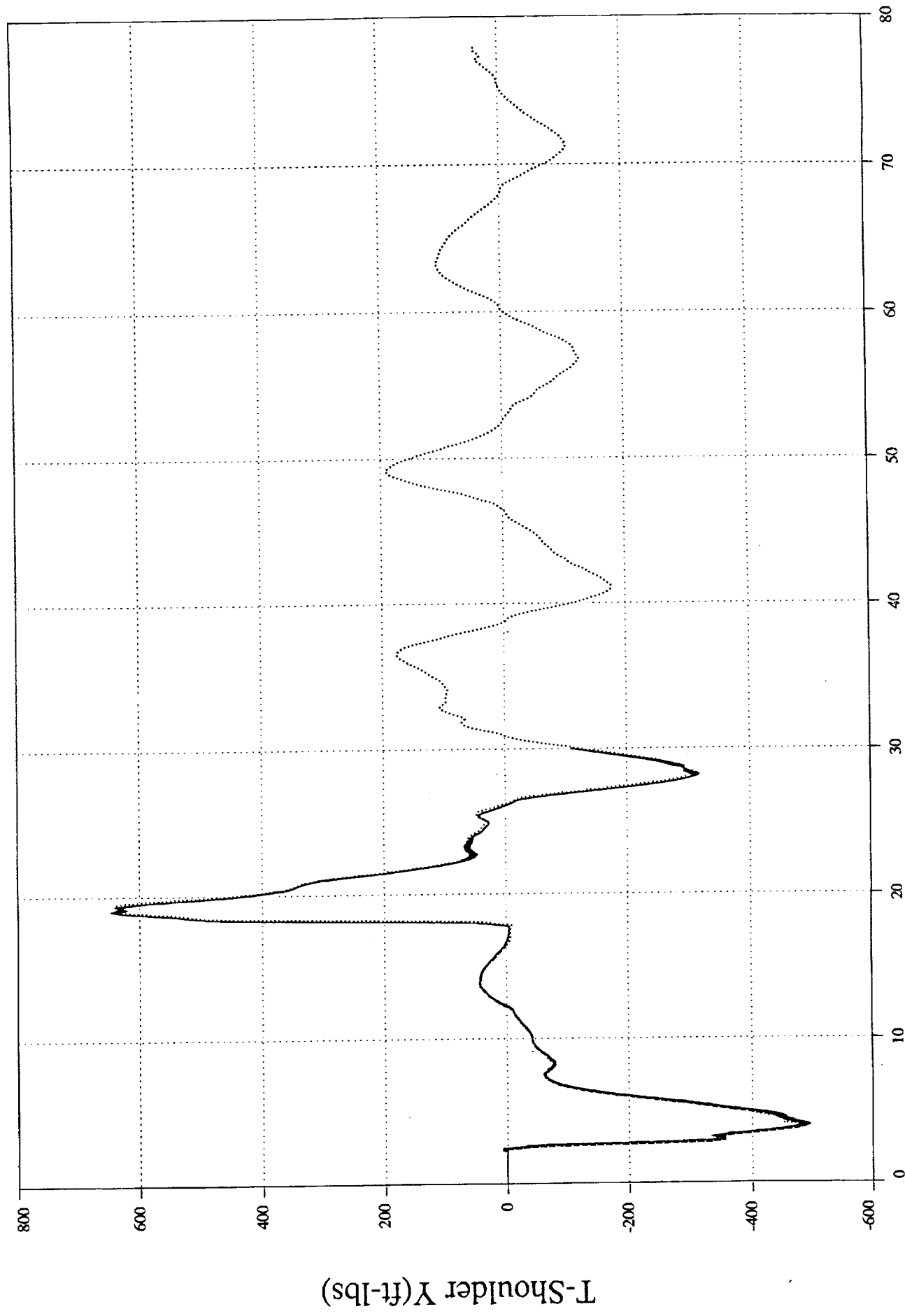


Figure 15  
TIME(sec)--->case27

val27      val27nac      SIMSGI-Shoulder Z Moment      v27hc

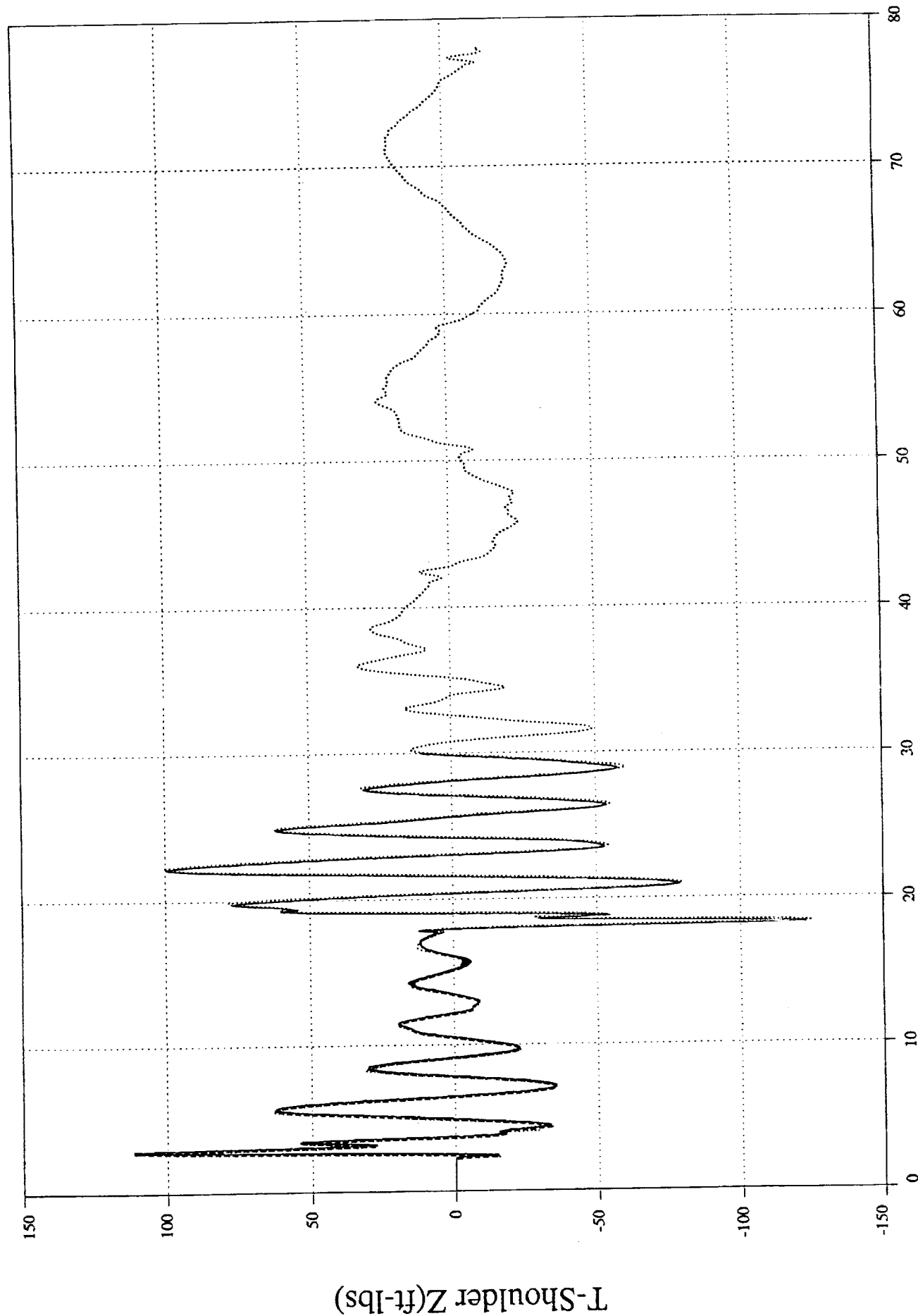


Figure 16

TIME(sec)--->case27

## Reference 6

DCI TM#101497-1  
ROCKET Flight-to-Sim Validation Runs  
October 14, 1997

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts - Dr. Patrick Tobbe  
Subject : *ROCKET* Flight to Simulation (FTS) Validation Runs

## Introduction

The real time Remote Manipulator System (RMS) simulation *ROCKET*, hosted on the SGI Challenge machine UQBAR, was modified to run a series of Flight to Simulation validation cases. The results of these runs were previously documented in Technical Memorandum #092997-1. In particular, an intermittent high frequency oscillation was discovered in the strain gage loads for some runs. This memorandum will describe software changes made to *ROCKET* to eliminate this effect.

## Software Modifications

As stated in the earlier memorandum, the high frequency oscillation present in *ROCKET* and ALLI data is driven by rigid body angular accelerations of the RMS booms. This effect was further traced to the coulomb friction model used for the RMS joints. Normally, the friction torque has a constant amplitude and opposes the joint velocity. To avoid a discontinuity at zero velocity, the friction torque is modeled as a straight line with a steep slope in this region. For some Flight-to-Simulation (FTS) validation runs, this slope was too steep for the integration step used. This phenomena has previously been recognized in contact tests and Simulation-to-Simulation validation runs. However, due to the data output rate used on the Alliant, this effect had gone unnoticed in the FTS results. To correct this problem, the coulomb friction model slope for the FTS runs was reduced to the value used for the STS and contact runs. This change was made in the subroutine *RMSPLANT*. The original code was as follows

```
IF ((VALIDATION_MODE.GE.1).AND.(VALIDATION_MODE.LE.30)) THEN
  EPS_FRIC_GB = EPSFRIC
ELSE
  EPS_FRIC_GB = 10.D0*J_GEAR_RATIO(I)*EPSFRIC
ENDIF
```

where EPSFRIC is a model parameter set in *DATABLOC*. EPS\_FRIC\_GB is the angular velocity at the maximum friction torque. For the STS and contact runs, this value was scaled by ten times the gear ratio. For the FTS runs, EPS\_FRIC\_GB is just EPSFRIC. The code was modified to use the scaled value of EPS\_FRIC\_GB for all cases.



## **Simulation Results**

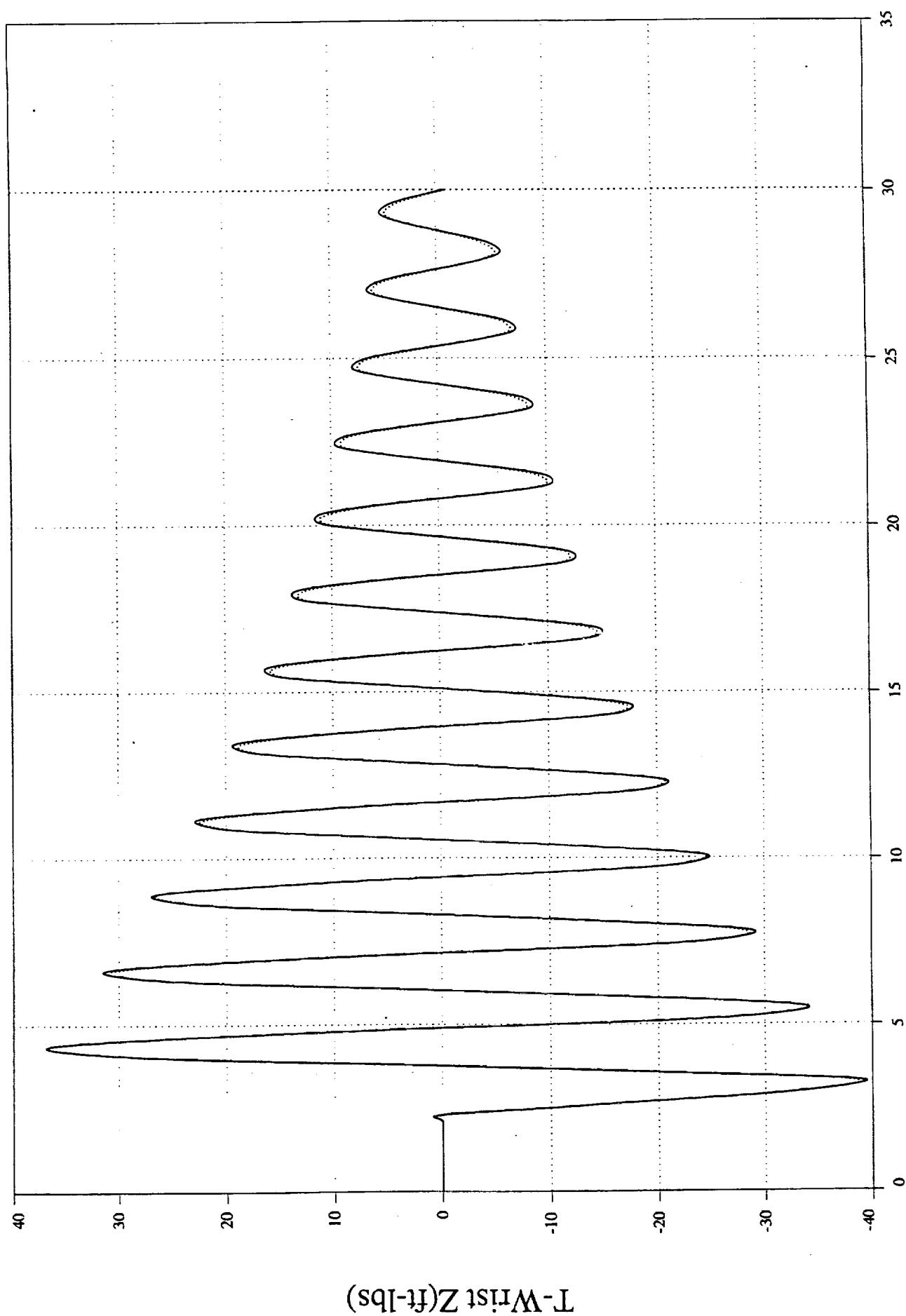
The twenty-nine FTS runs were repeated with the modified friction parameter. The results of four selected cases are shown in Figures 1 through 16 with data from the original ALLI FTS runs. The loads using the scaled friction parameter are close to previous values and do not significantly change validation results. The high frequency oscillations are no longer present as the new results are smoother.

## **Conclusions / Recommendations**

The slope of the coulomb friction model was modified for the FTS runs to eliminate high frequency oscillations in the RMS member internal loads. Simulation results do not significantly alter FTS validation results. The resulting software for the modified friction model is also simpler than the previous version as all conditions now use the same value of the friction parameter.

10—033





TIME(sec)--->case01

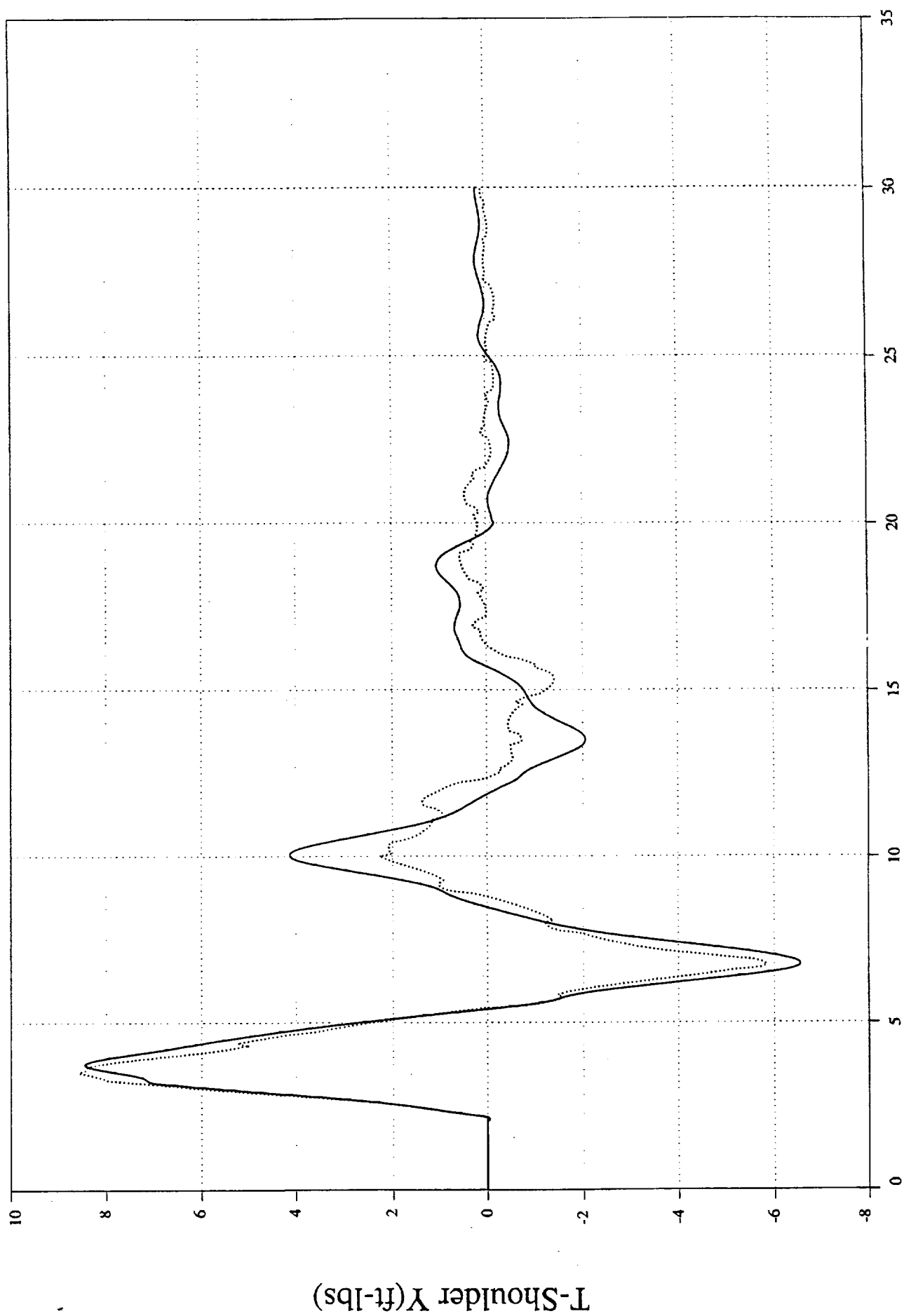
Figure 2

SIMSGI-Shoulder Y Moment

pl002

1 0 39

1 0 19

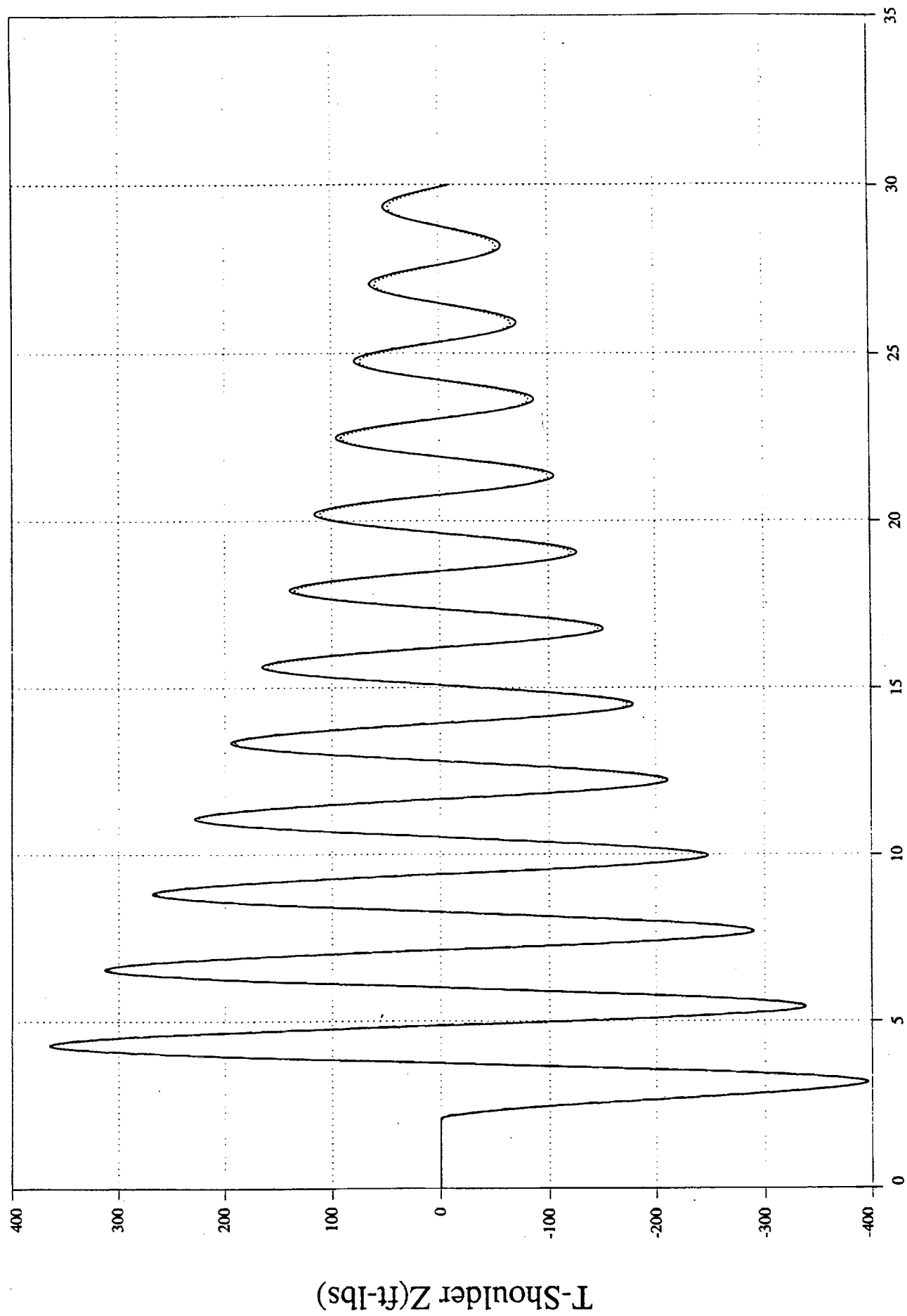


TIME(sec)--->case01

Figure 3

SIMSGI-Shoulder Z Moment  
1 0 ..... 20

p1002  
1 0 — 40



TIME(sec)--->case01

Figure 4

SIMSGI-Wrist Y Moment

t/././fsva/v8hc

p8002

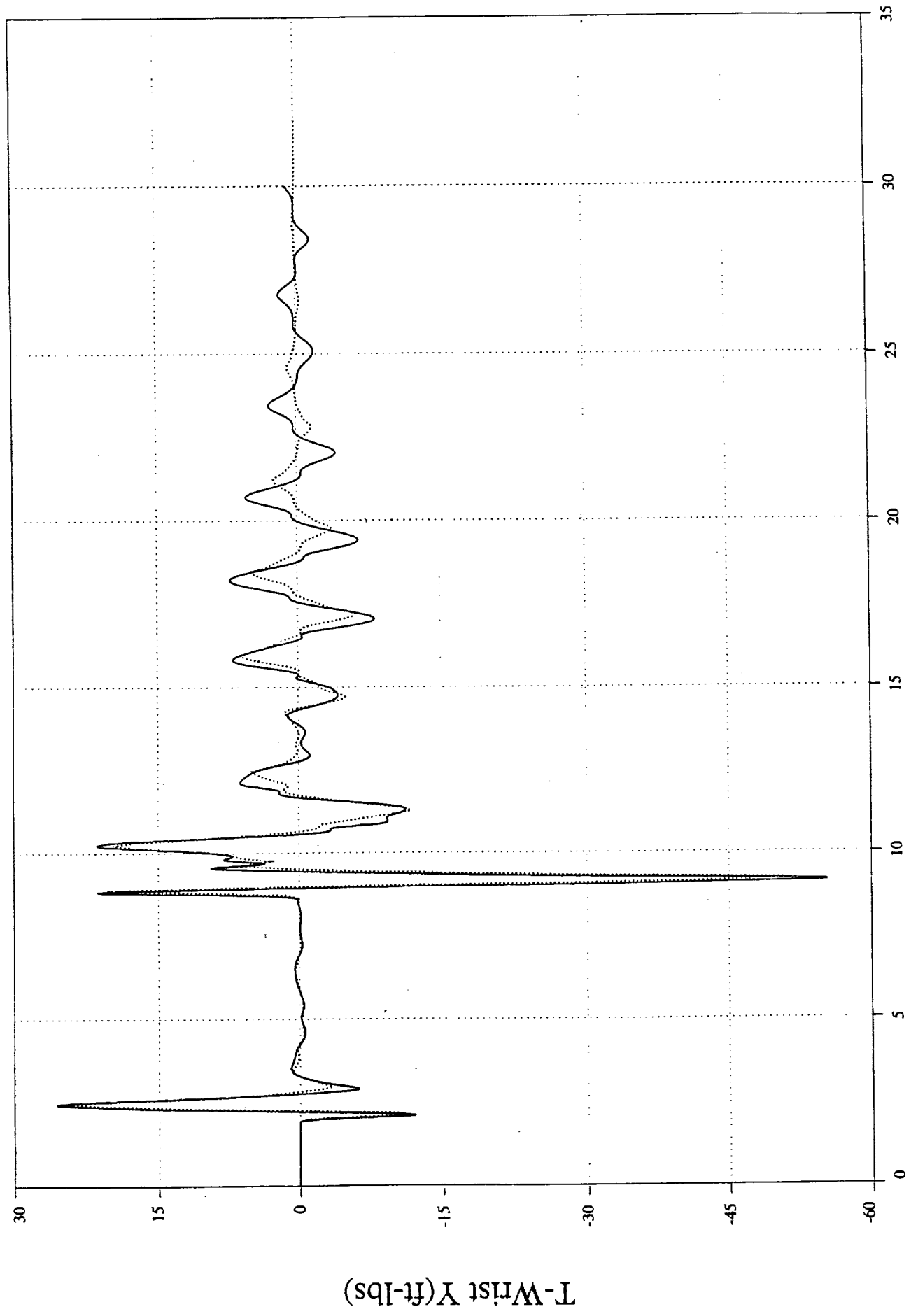
1

1

25

33

35



TIME(sec)--->case08

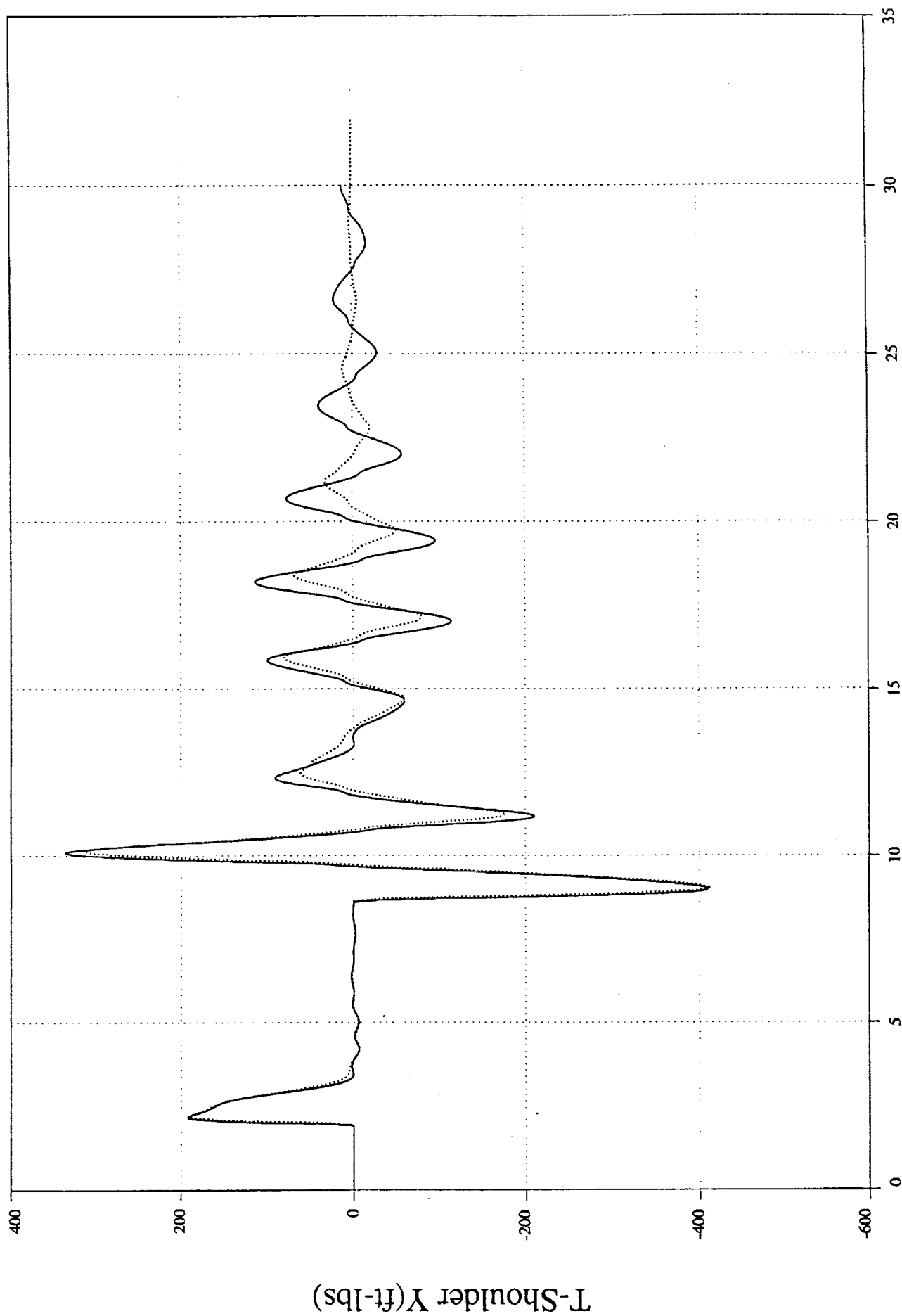
Figure 5



SIMSGI-Shoulder Y Moment

p8002

1 0 39



TIME(sec)--->case08

Figure 7

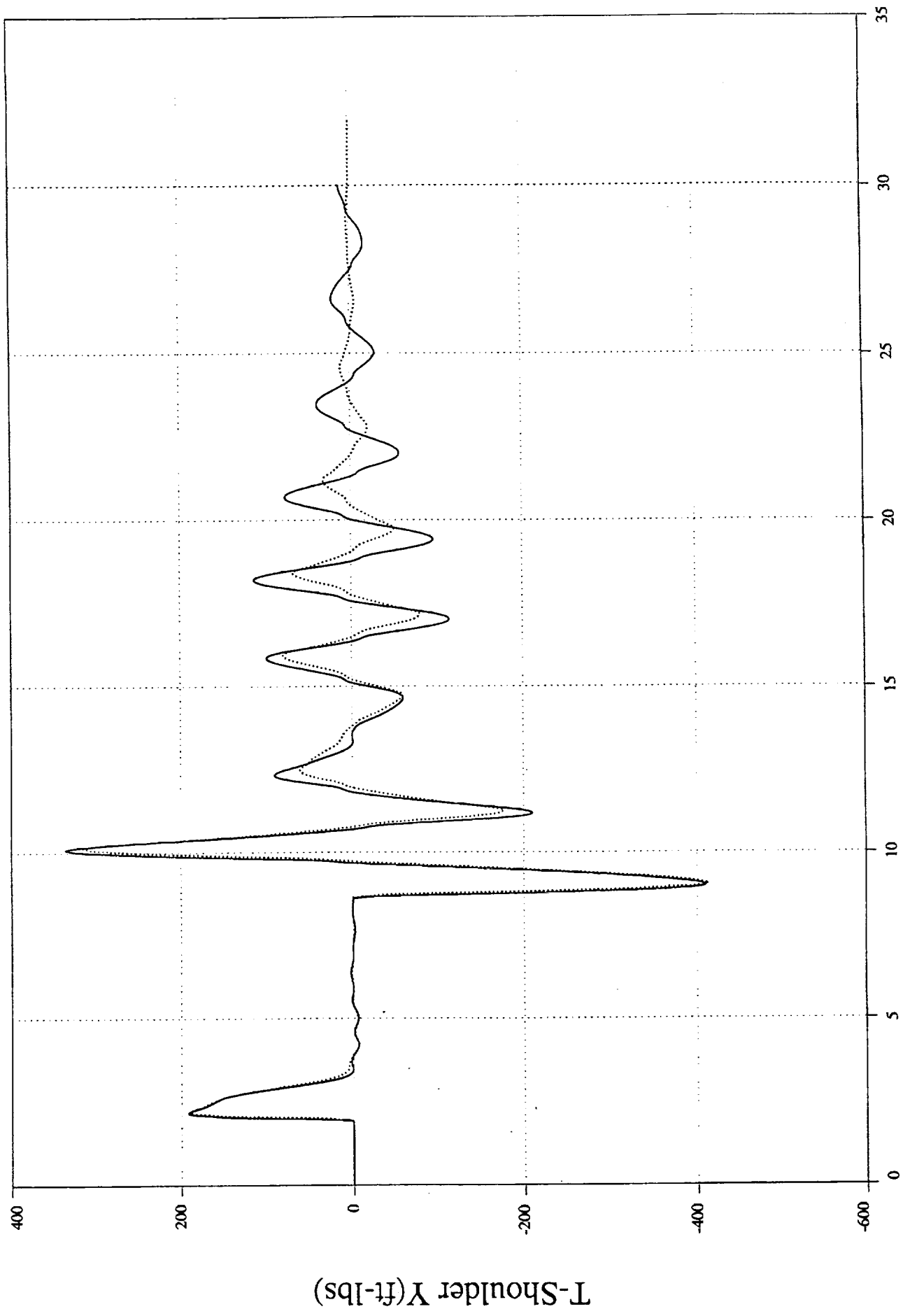


SIMSGI-Shoulder Y Moment

p8002

1

19



TIME(sec)--->case08

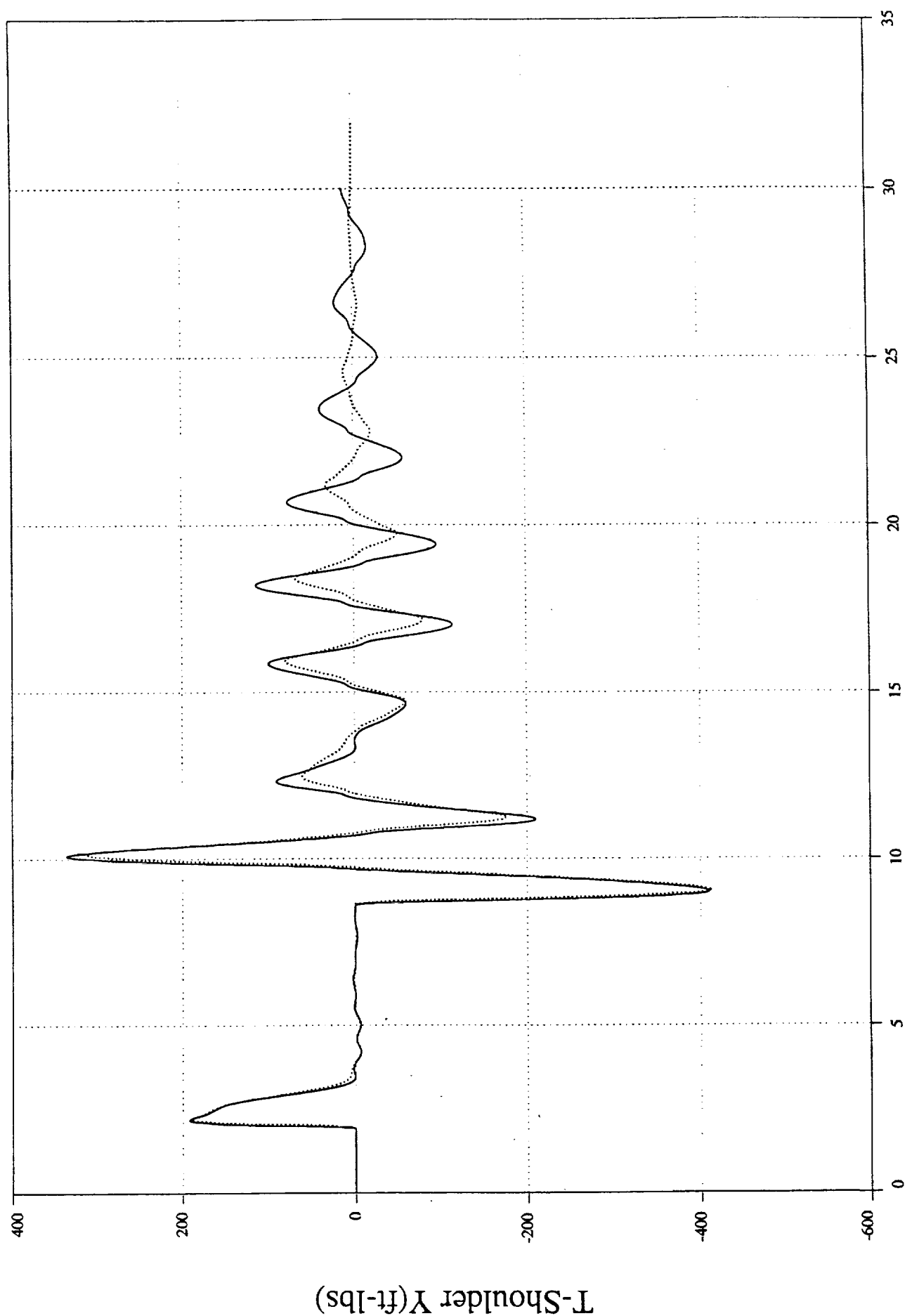
Figure 7

SIMSGI-Shoulder Y Moment

p8002

1 0 39

1 0 19



TIME(sec)--->case08

Figure 7

SIMSGI-Shoulder Y Moment

1 □ ..... 19

p8002

1 ○ ——— ○ 39

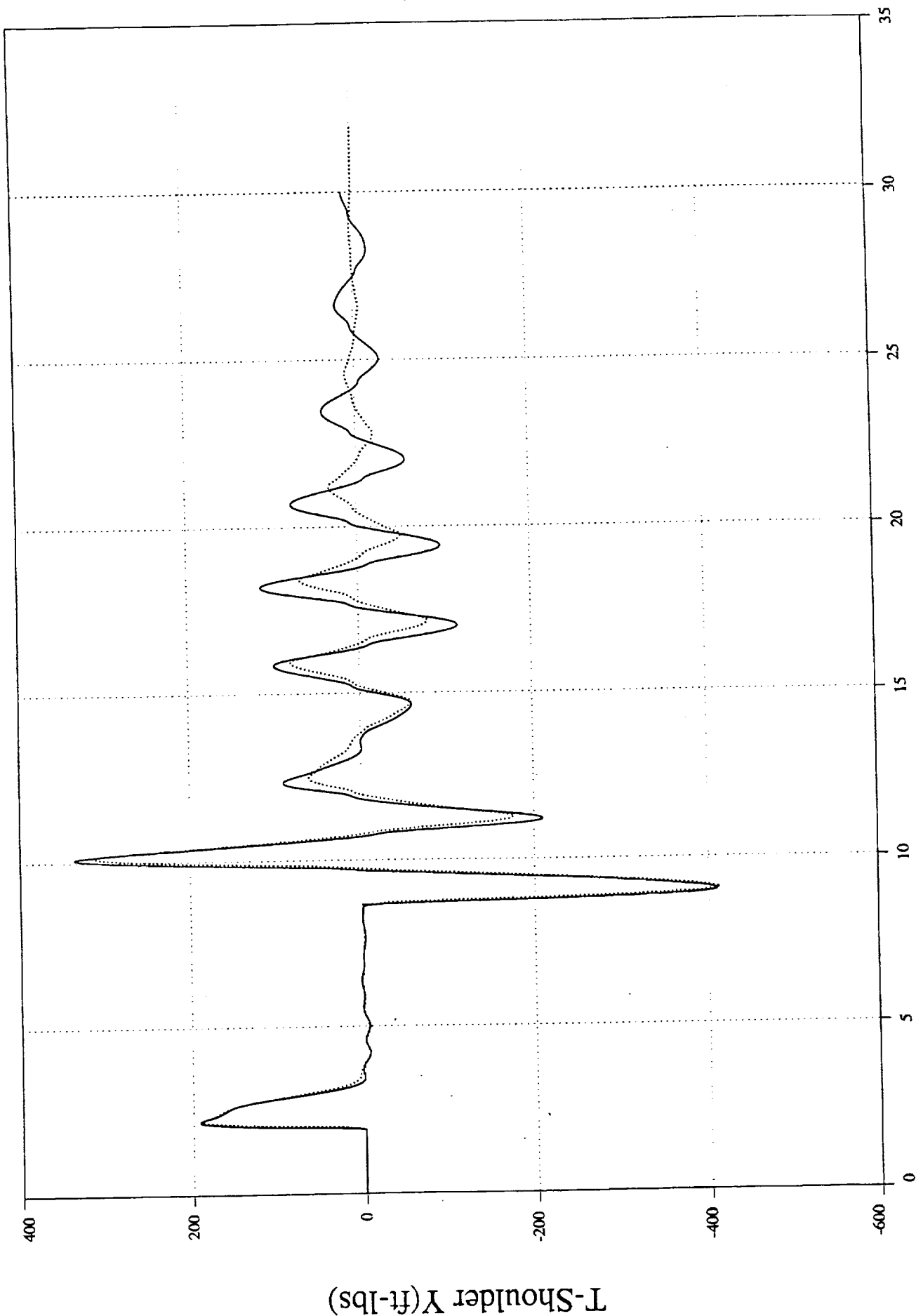


Figure 7

SIMSGI-Shoulder Z Moment  
p8002  
1 0 40  
1 0 20

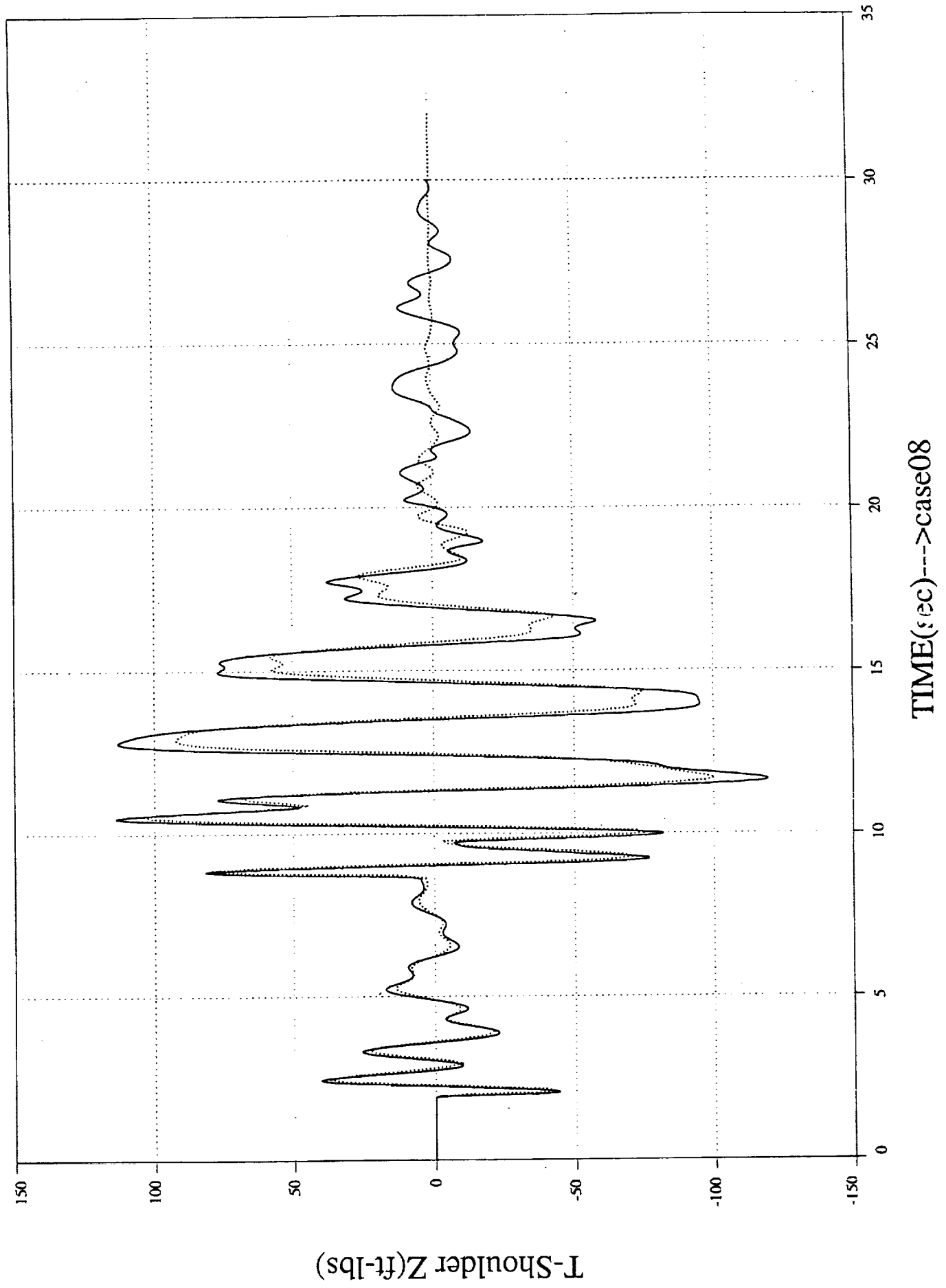


Figure 8

SIMSGI-Wrist Y Moment

p23002

1 ○ — ○ 33

1 □ ..... □ 25

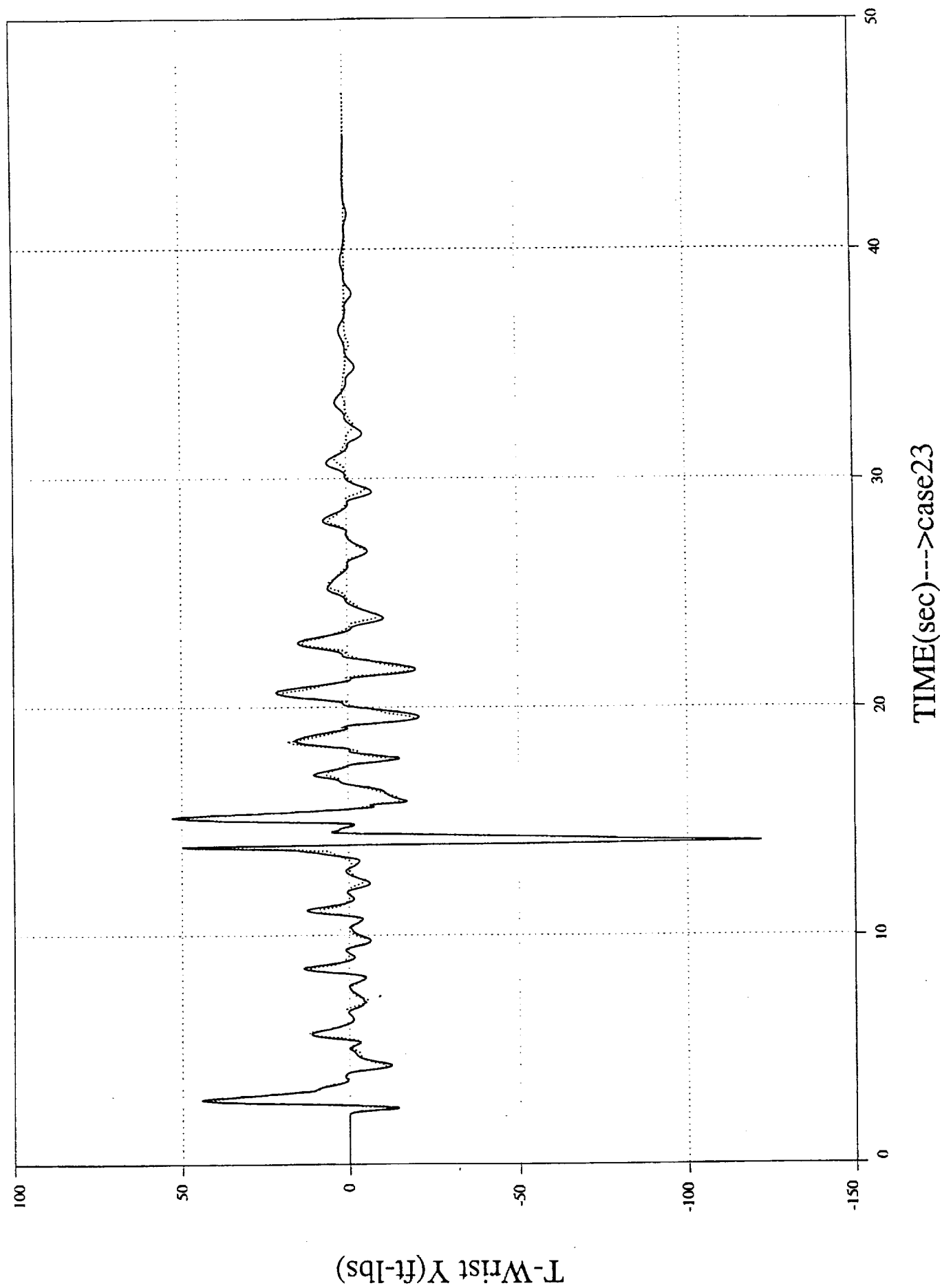


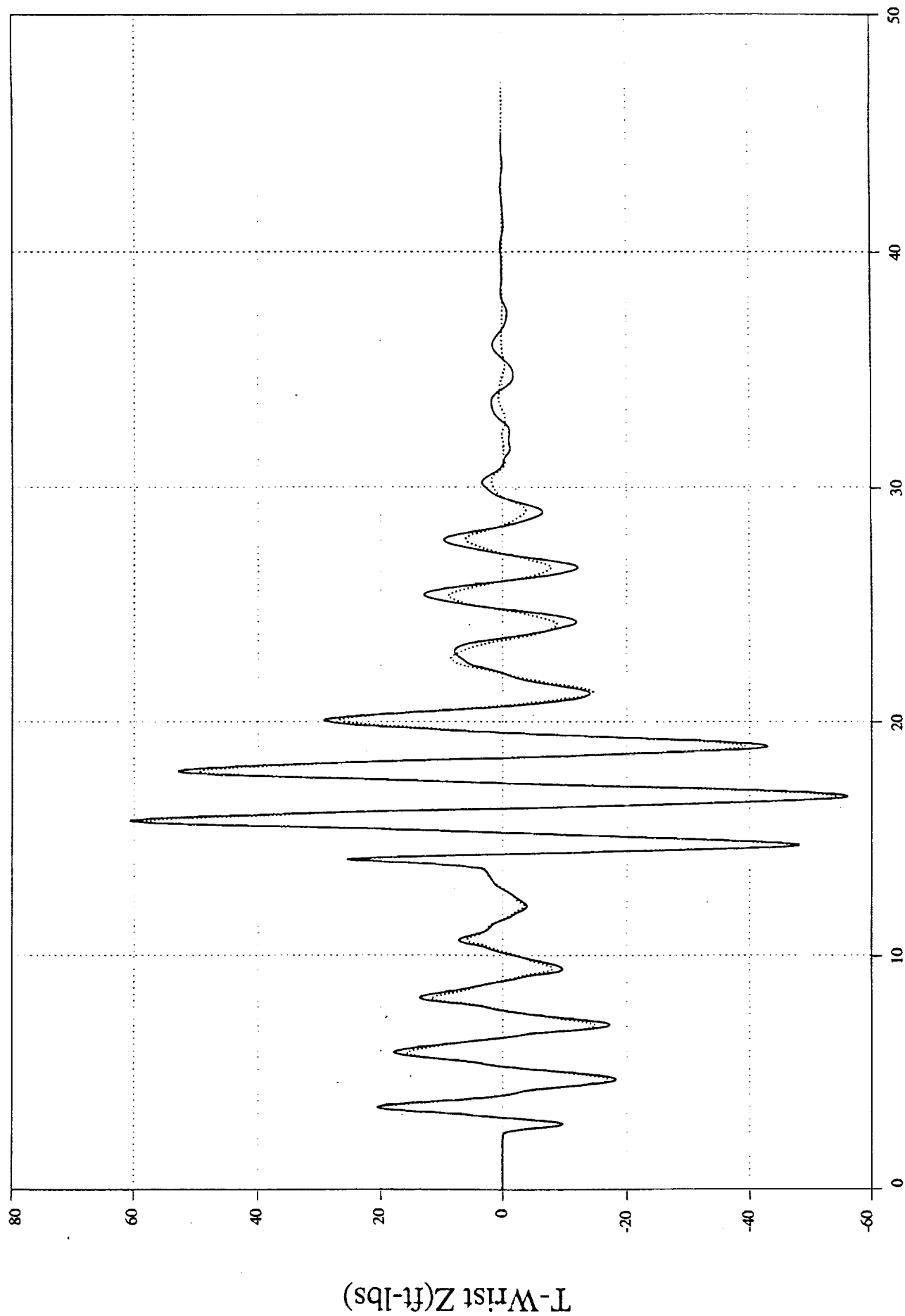
Figure 9

p23002

1 0 — 34

SIMSGI-Wrist Z Moment

1 0 — 26



TIME(sec)--->case23

Figure 10

SIMSGI-Shoulder Y Moment

p23002

fsval/v23hc

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

19

39

1

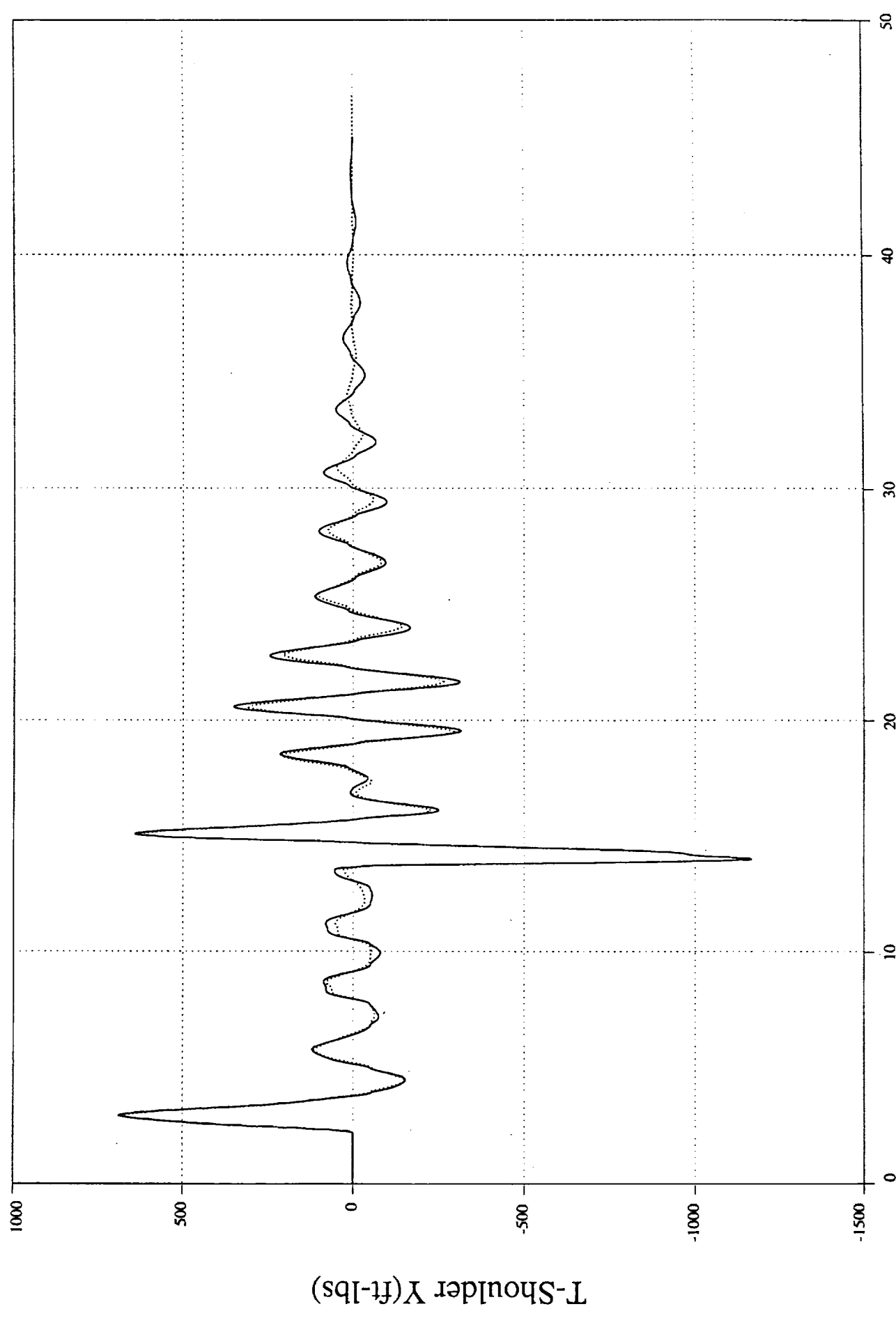
19

39

1

19

39



TIME(sec)--->case23

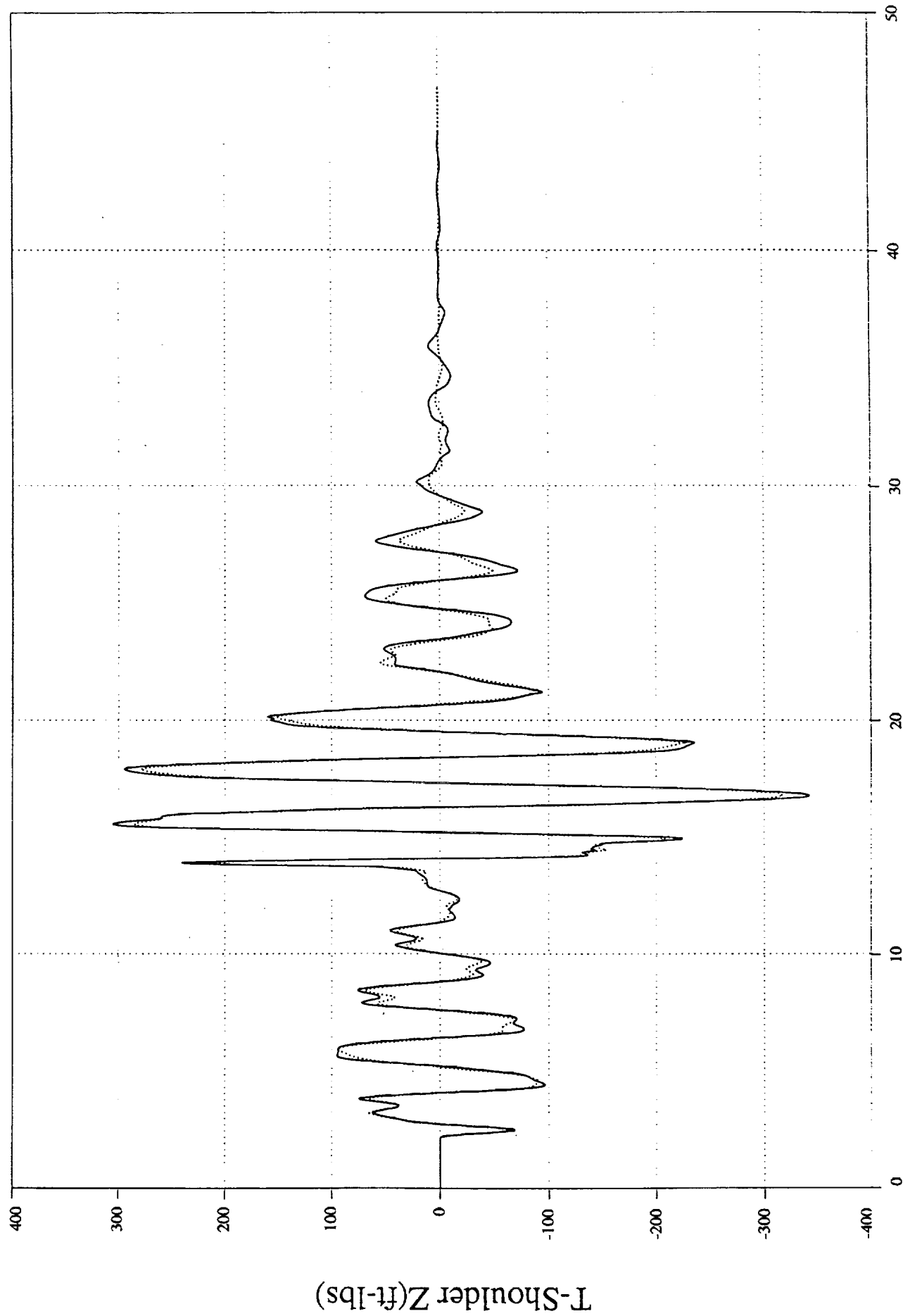
Figure 11

p23002

SIMSGI-Shoulder Z Moment

1 □ ..... □ 20

1 ○ ..... ○ 40



TIME(sec)--->case23

Figure 12



SIMSGI-Wrist Y Moment<sup>1</sup>.../fsval/v27hc

1 □ ..... □ 25

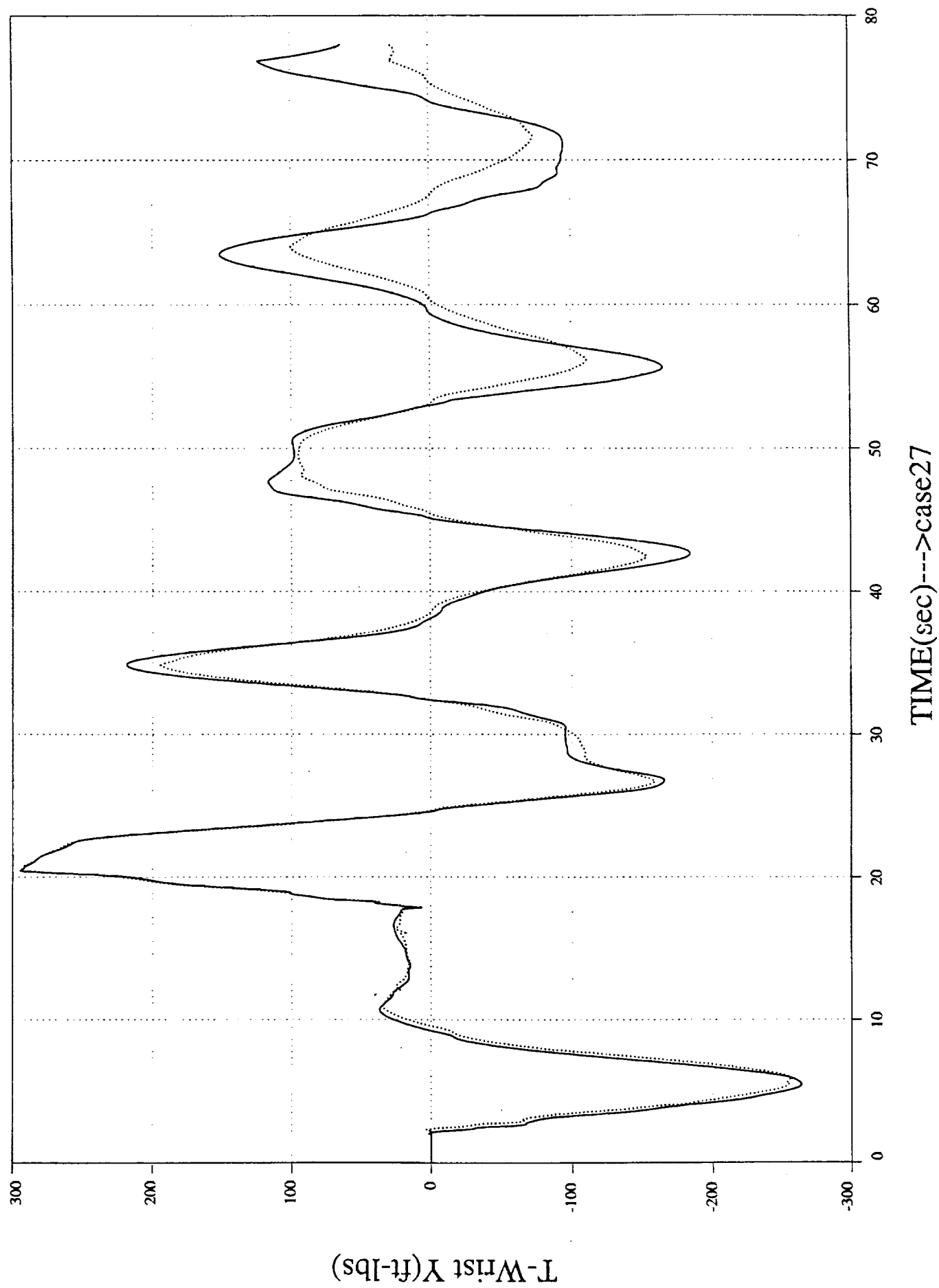


Figure 13

p27002

SIMSGI-Wrist Z Moment

1 0 34

1 0 26

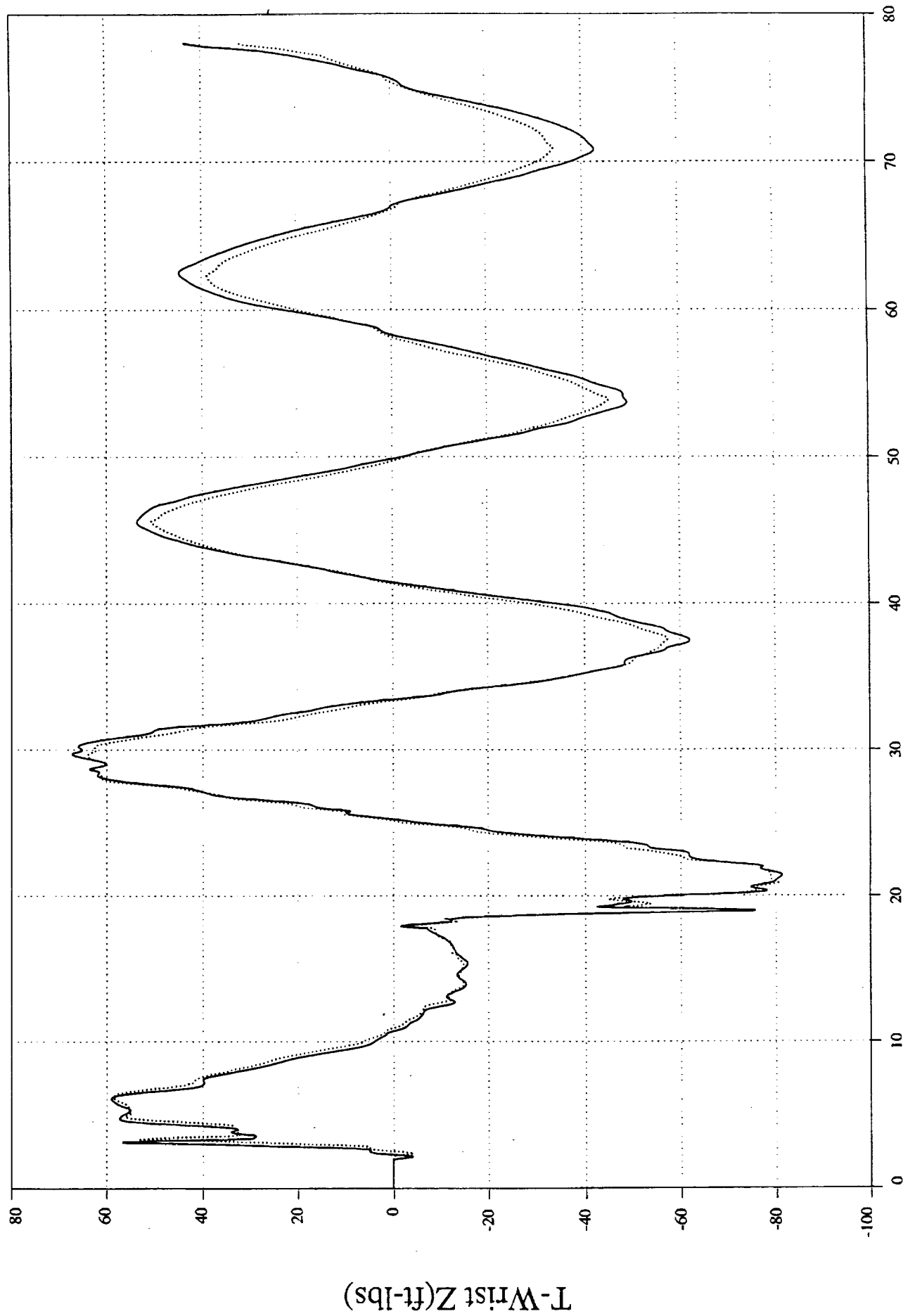


Figure 14

SIMSGI-Shoulder Y Moment  
1 19

p27002  
1 39

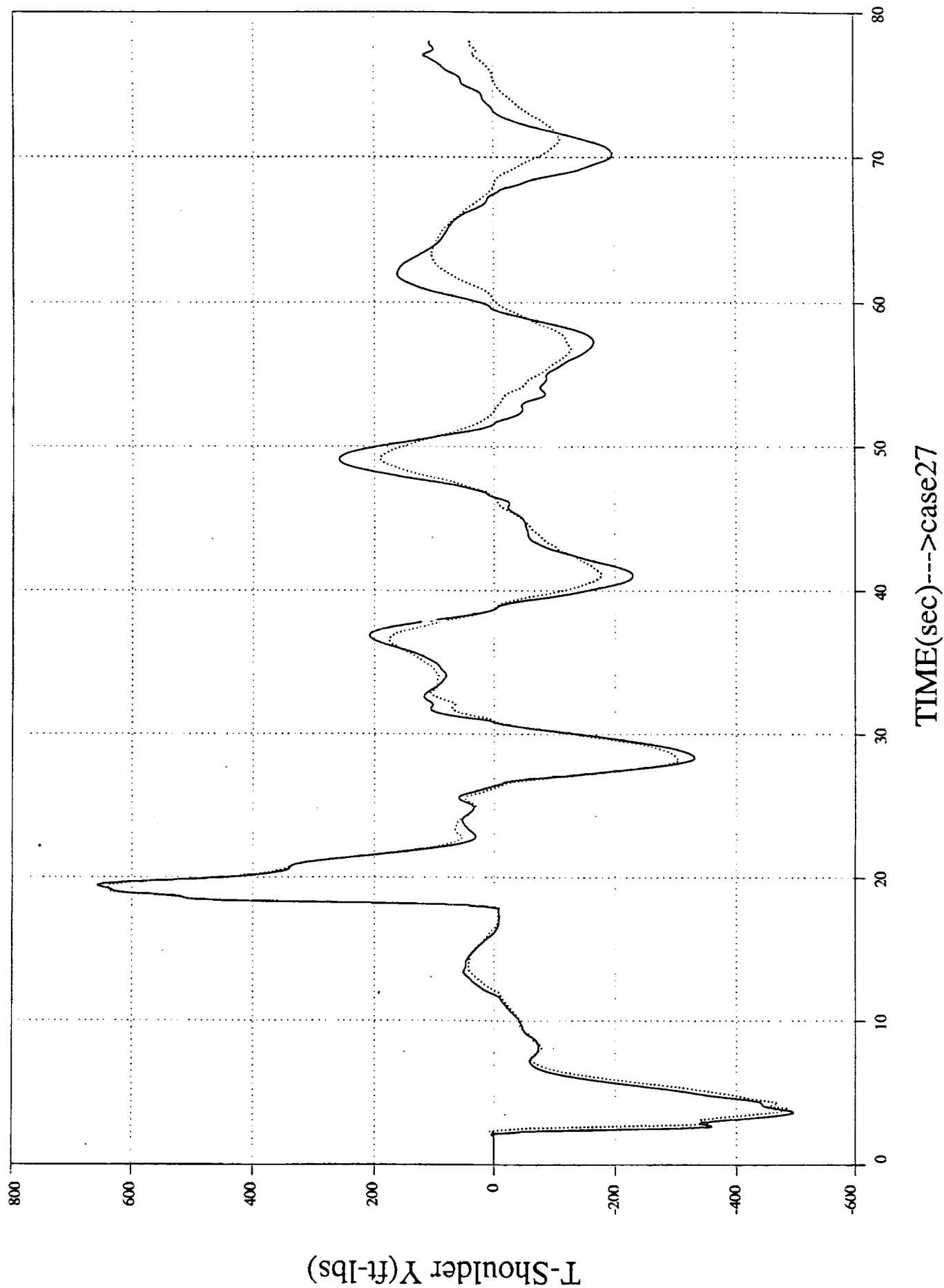


Figure 15

SIMSGI-Shoulder Z Moment

p27002

#fsval/v27hc

1 □ ..... □ 20

1 ○ ..... ○ 40

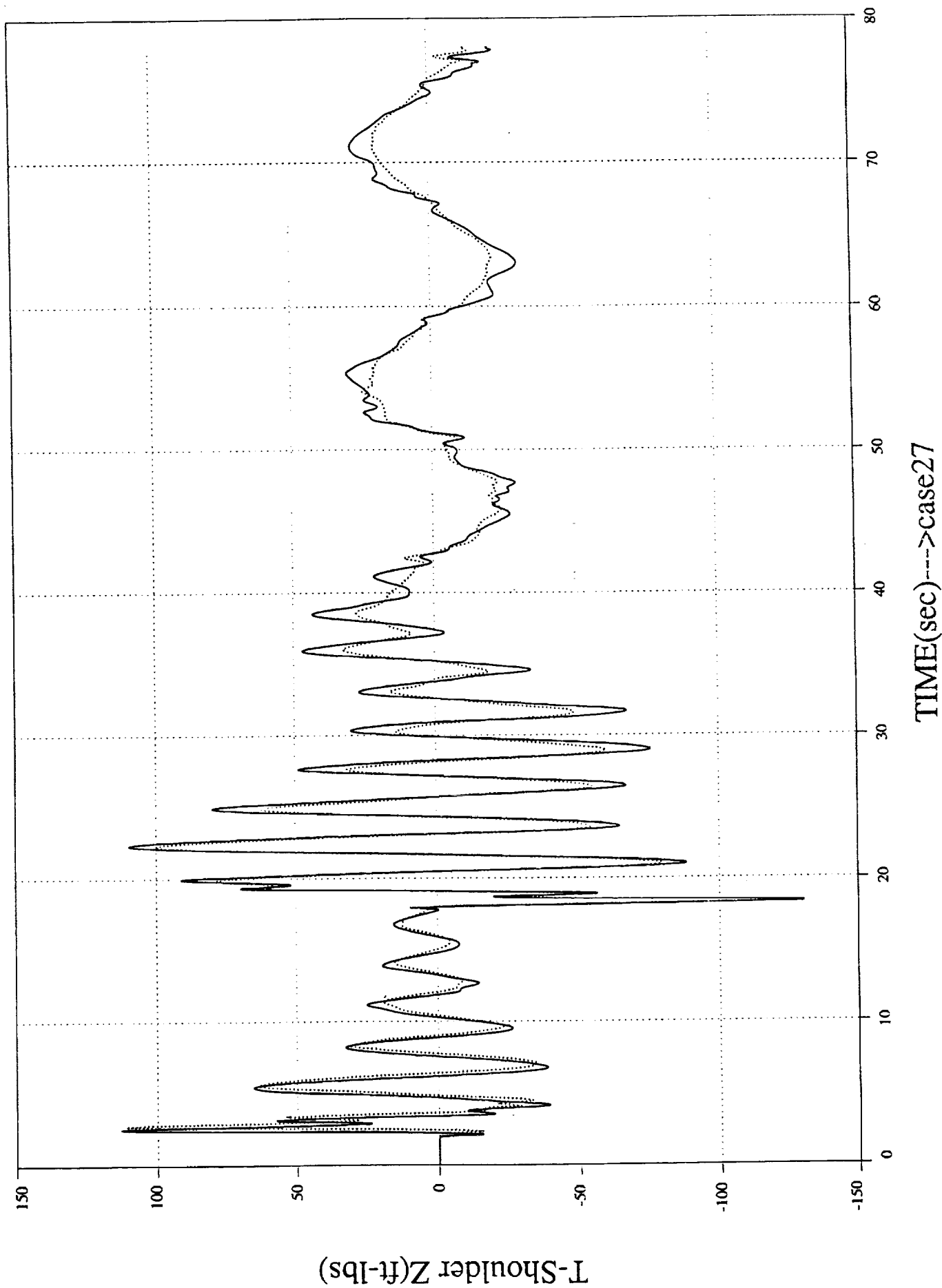


Figure 16

## Reference 7

### Bridge Encoder Test Report October 7, 1997

Test Report: Bridge Encoder Test Report  
October 7, 1997

Marlin J. Williamson, bd Systems

**Objective:** Independently measure changes in position of the DOTS tip position while exercising only the bridge joint. Verify the encoder scale factor used in the DOTS software to convert raw encoder data to engineering units.

**Test Setup:** The DOTS was positioned so that the payload was pointed west with the its face somewhat perpendicular to the floor. The alignment was accomplished through jog panel moves. The joint angles and tip position was observed on the display. (The last encoder calibration was performed 8/27/96, the zero locations were checked 9/19/97)

A laser range finder was placed upon the platform within the stationary three point docking mechanism. The DOTS was positioned so that the lasers spot from the range finder reflected off of a hard surface on the DOTS payload. A sheet of highly reflective material was taped on the payload and us to reflect the range finder laser.

The DOTS was then repositioned using only the bridge joint from the jog panel. The DOTS was moved to the east end of the floor. The laser range finder was repositioned so that its spot remained on the sheet of reflective material.

**Test Results:** Data recording was started from the DOTS with the terminal command *RECORD START BRIDGE*. Data was observed and recorded by hand from the range finder's display. The data recorder appears in the table below.

**Bridge Encoder Resolution Test Results**

| Bridge Joint (feet) | Range Finder Output (Meters) |
|---------------------|------------------------------|
| 65.99               | 17.527                       |
| 55.99               | 14.482                       |
| 45.99               | 11.429                       |
| 35.99               | 8.387                        |
| 25.99               | 5.335                        |
| 15.99               | 2.295                        |

The bridge joint was repositioned every 10 feet and the laser range finder output was recorded. The difference in position as measured from the range finder was as follows:

### Changes in Position Results

| Position Change     | Range Finder Output (Meters) |
|---------------------|------------------------------|
| From 65.99 to 55.99 | 3.045                        |
| From 55.99 to 45.99 | 3.053                        |
| From 45.99 to 35.99 | 3.042                        |
| From 35.99 to 25.99 | 3.052                        |
| From 25.99 to 15.99 | 3.040                        |

Using the conversion factor: 3.280839895 feet/meters (CRC Standard Mathematical Tables, 23<sup>rd</sup> Edition, and Page 4), 10 feet equates to 3.048 meters. The largest error from the above table is only 0.2% which could be contributed by the positioning method. It is much less than the 3% error between the DOTS tip position and the VGS position that was observed during the VGS DOTS motion testing.

To further investigate the potential for error, the data recorded by the DOTS was plotted using the same M-file for Matlab that was used for the VGS DOTS motion tests post processing.

The tip x position, encoder position, and laser range finder results are plotted. The first plot shows the bridge encoder output in feet. The computed tip x position of the DOTS payload is shown in the second plot (plotted in meters). The tip x position was computed in meters as so to directly compare with the output of the VGS.

The third plot show both the bridge encoder (now in meters) and the tip x position. The bridge encoder time history is shifted by an offset (so that it is equal to the tip x position at the start of the data) for the fourth plot. The fourth plot shows both the offset bridge encoder time history and the tip x position time history. They appear to lie on top of each other. The fifth plot is the error between the offset bridge encoder signal and the tip x position.

The sixth plot shows the laser range finder output plotted against fictitious time points. These times were obtained from the tip x position plots. There

were obtained for use in the seventh plot. This plot is of the range finder output and an offset tip x position. The curves appear to be identical during period of zero position changes. The offset was selected so that the tip x position would equal the range finder output at the first data point.

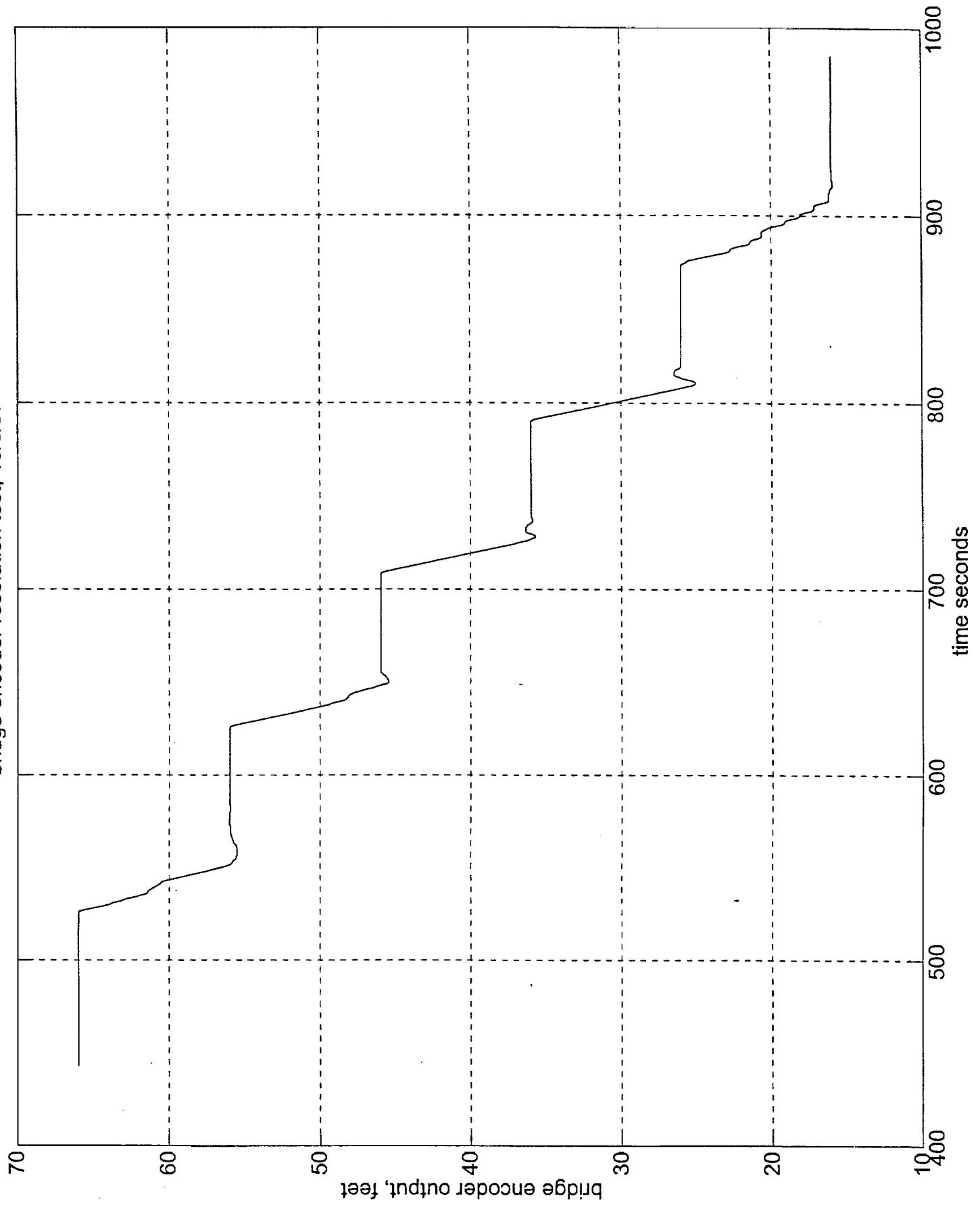
The offsets were necessary because the coordinate frames were not set up prior to the data collection. An error of magnitude 3 % was not observed in the encoder readings or in the computed data (tip position) used to compare to the VGS data.

This test was performed after the bridge encoder coupler was tightened.

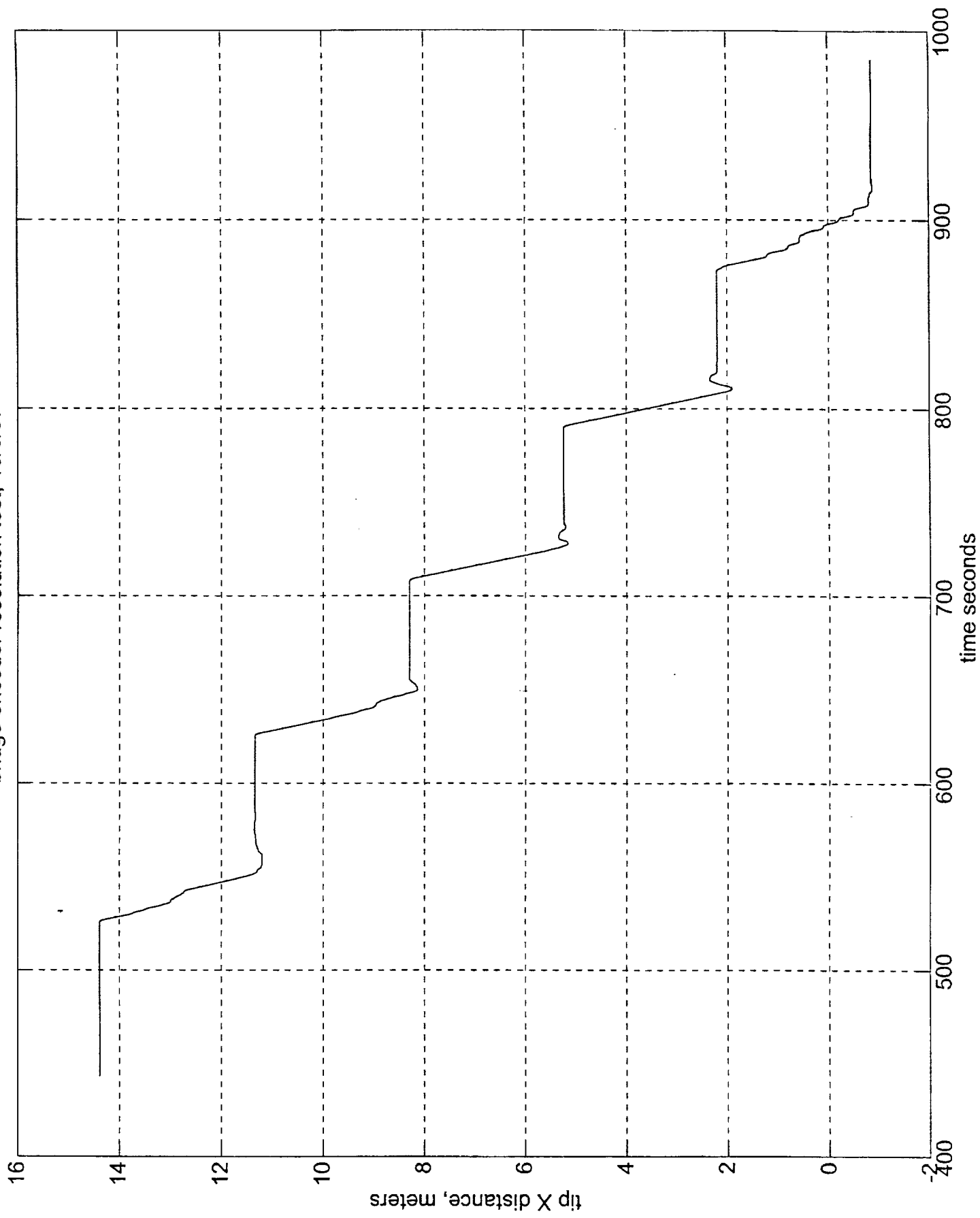
Enclosed are also notes from the encoder zero check of 9/19/97 and the M-file used to post process the binary data collected by the DOTS.



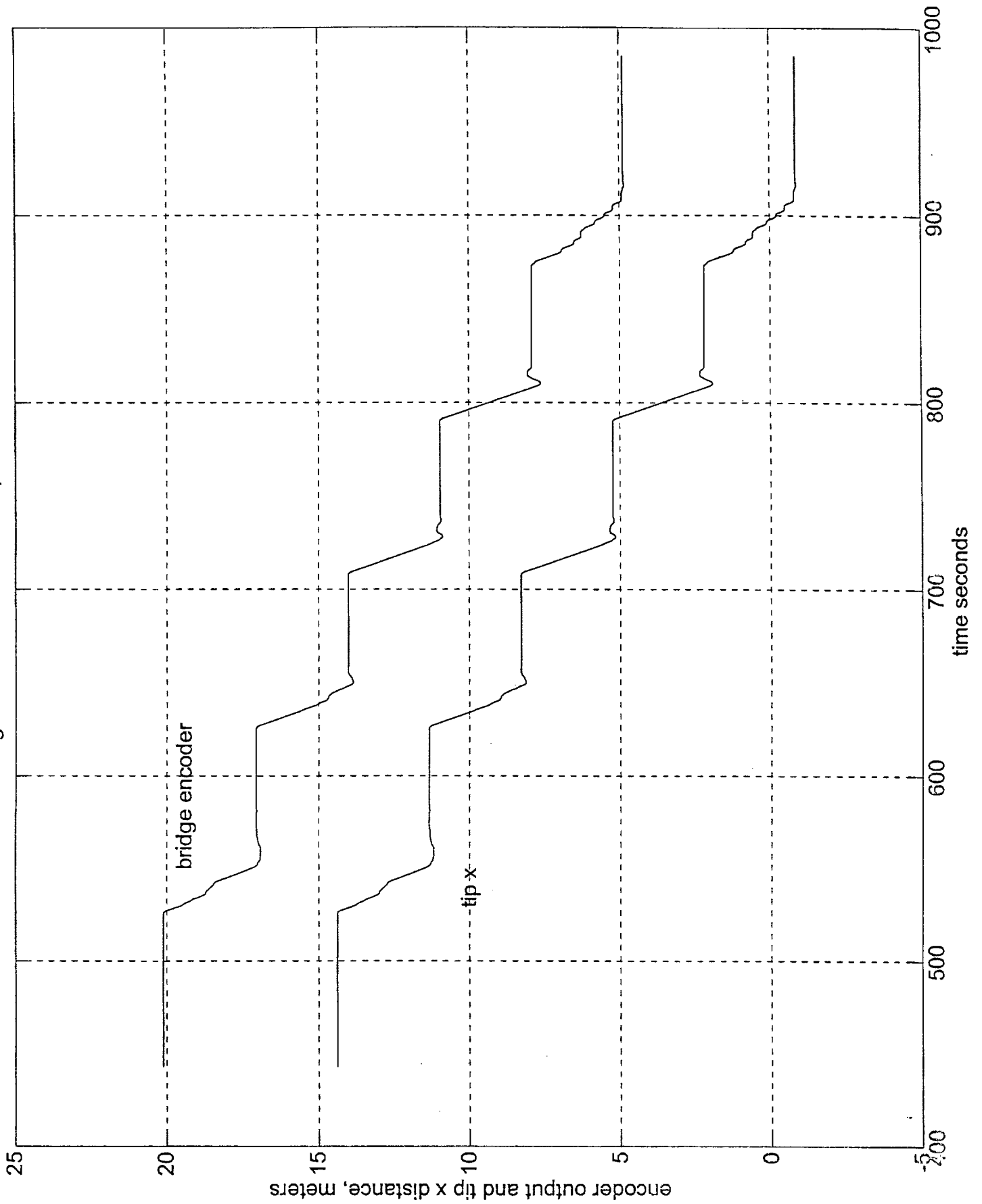
bridge encoder resolution test, 10/6/97



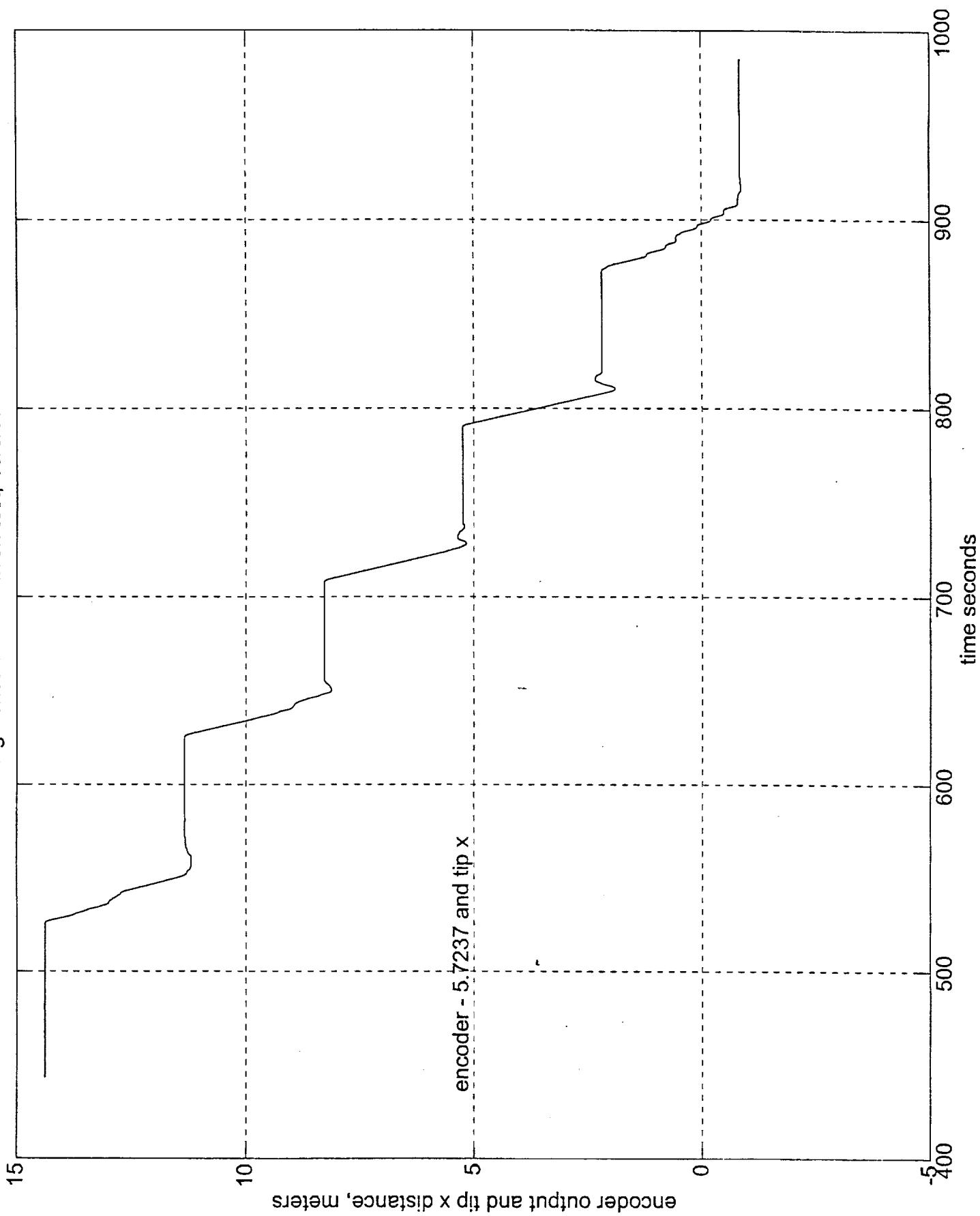
bridge encoder resolution test, 10/6/97



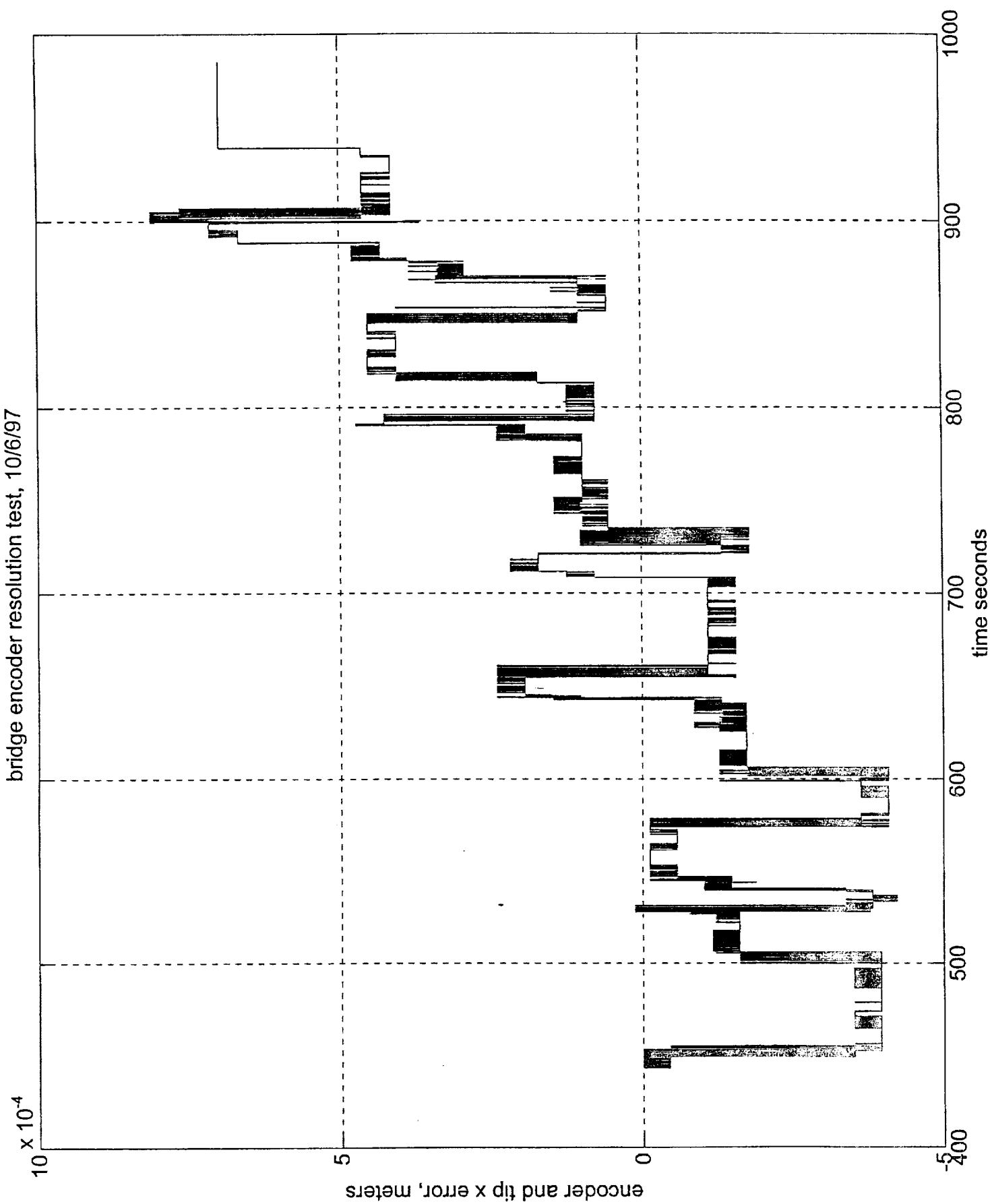
bridge encoder resolution test, 10/6/97



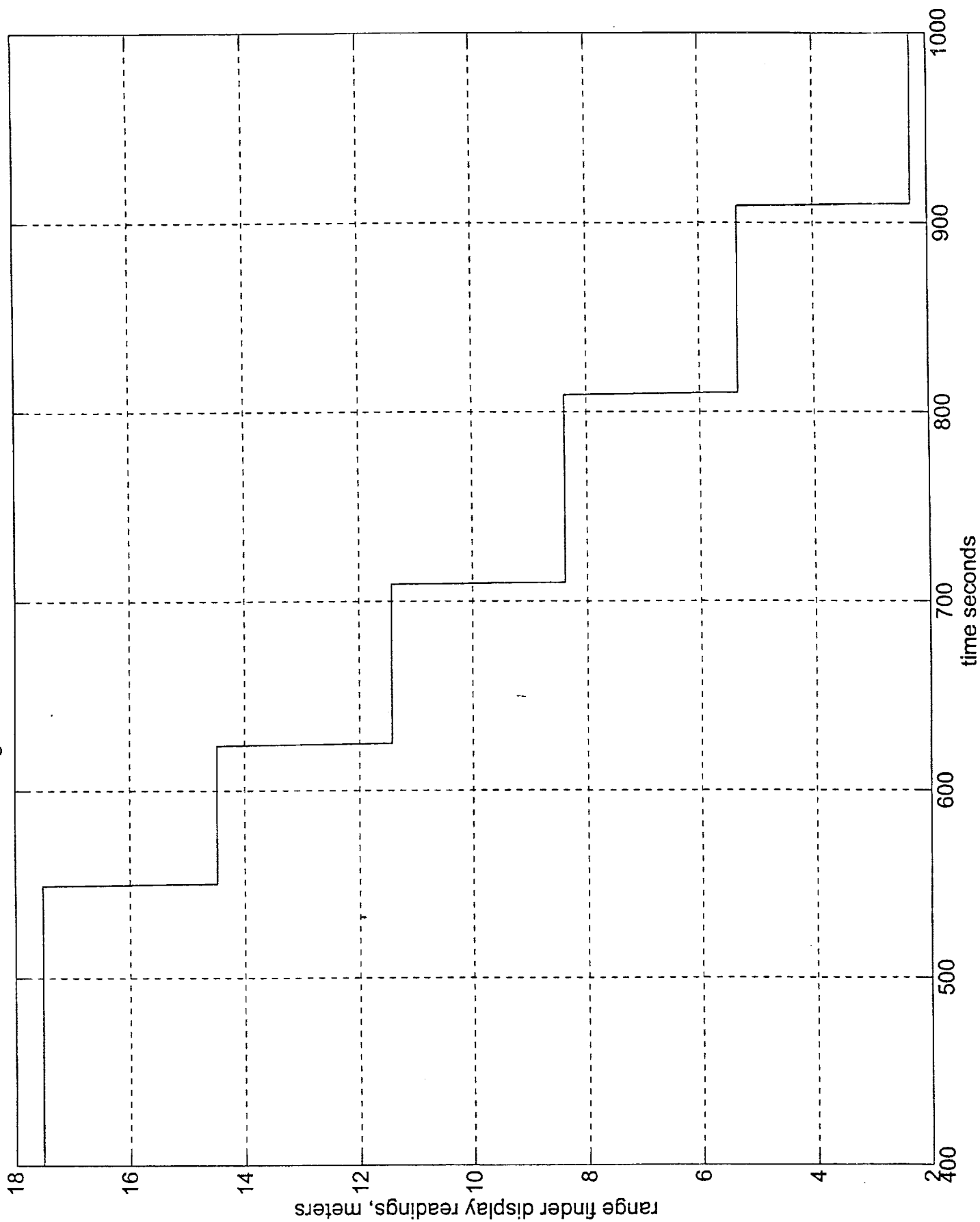
bridge encoder resolution test, 10/6/97



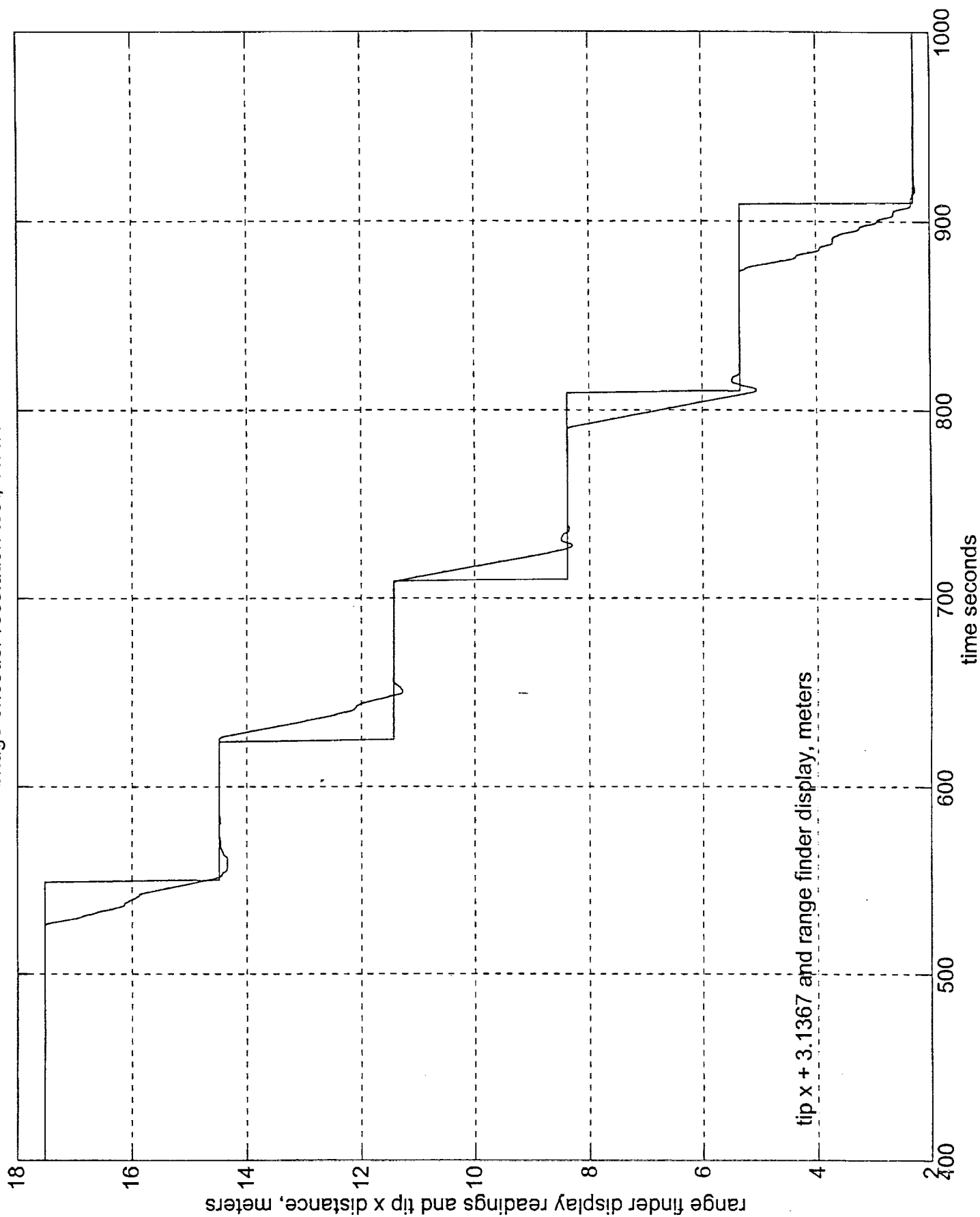
bridge encoder resolution test, 10/6/97



bridge encoder resolution test, 10/6/97



bridge encoder resolution test, 10/6/97



tip x + 3.1367 and range finder display, meters

```
% Matlab input files for use with DOTS output (VGS Testings) 5/97
% Ask for a test prefix, then reads in binary data from DOTS output files
```

```
run = input('Enter a run prefix: ','s');
```

```
filename1 = [run '.tac'];
fid=fopen(filename1,'r','b');
if fid >=0
    tach=fread(fid,[9,inf],'double');
else
    msg=['Tachometer Output File Not Found']
end
```

```
filename1 = [run '.enc'];
fid=fopen(filename1,'r','b');
if fid >=0
    encode=fread(fid,[8,inf],'double');
else
    msg=['Encoder Output File Not Found']
end
```

```
filename1 = [run '.srv'];
fid=fopen(filename1,'r','b');
if fid >=0
    servo=fread(fid,[8,inf],'double');
else
    msg=['Servo Voltage Output File Not Found']
end
```

```
filename1 = [run '.jcm'];
fid=fopen(filename1,'r','b');
if fid >=0
    joint=fread(fid,[8,inf],'double');
else
    msg=['Joint Command Output File Not Found']
end
```

```
filename1 = [run '.tcm'];
fid=fopen(filename1,'r','b');
if fid >=0
    tipcom=fread(fid,[6,inf],'double');
else
    msg=['Tip Command Output File Not Found']
end
```

```
filename1 = [run '.tax'];
fid=fopen(filename1,'r','b');
if fid >=0
    tipact=fread(fid,[6,inf],'double');
else
```



```

    msg=['Tip Actual Output File Not Found']
end

filename1 = [run '.slr'];
fid=fopen(filename1,'r','b');
if fid >=0
    solar=fread(fid,[11,inf],'double');
    ss_act=solar(1,:);
    ss_go=solar(2,:);
    ss_x=solar(3,:);
    ss_theta=solar(4,:);
    ss_status=solar(5,:);
    ss_ang=solar(6,:);
    ss_angdot=solar(7,:);
    ss_cycle=solar(8,:);
    ss_target=solar(9,:);
    ss_tarx=solar(10,:);
    ss_tary=solar(11,:);
else
    msg=['Solar Output File Not Found']
end

filename1 = [run '.tim'];
fid=fopen(filename1,'r','b');
if fid >=0
    time=fread(fid,[1,inf],'float');
else
    msg=['Time Output File Not Found - BiG Problem']
end

filename1 = [run '.vgs'];
fid=fopen(filename1,'r','b');
if fid >=0
    vgs=fread(fid,[12,inf],'float');
    vgs_status=vgs(1,:);
    vgs_run=vgs(2,:);
    vgs_x=vgs(3,:);
    vgs_y=vgs(4,:);
    vgs_z=vgs(5,:);
    vgs_q0=vgs(6,:);
    vgs_q1=vgs(7,:);
    vgs_q2=vgs(8,:);
    vgs_q3=vgs(9,:);
    vgs_th1=vgs(10,:);
    vgs_th2=vgs(11,:);
    vgs_th3=vgs(12,:);
else
    msg=['VGS Output File Not Found']
end

i231=[ 2, 3, 1 ];
n=max(size(time));

```

```

d2pd2=[-1 0 0;0 -1 0; 0 0 1];
for ii=1:n,

    cm1=eul2(tipact(4,ii)*180/pi);
    cm2=eul3(tipact(5,ii)*180/pi);
    cm3=eul1(tipact(6,ii)*180/pi);
    cm=cm3*cm2*cm1;

    temp=eafa(i231,cm);
    ea_231(:,ii)=temp';
    dcm=d2pd2*cm;
    qtemp=qfa(dcm);
    dots_q(:,ii)=qtemp';
    temp=eafa(i231,dcm);
    y_231(:,ii)=temp';
    dots_th1=y_231(1,:)*pi/180;
    dots_th2=y_231(2,:)*pi/180;
    dots_th3=y_231(3,:)*pi/180;

%   account for 180 shift in yaw by negative signs on pitch and roll

end

```

# Encoder Notes

encoder calibration dates

9/19/97 (check only no changes)

8/27/96

<No calibration prior to V&S Testing>

Bridge

$$\begin{array}{r} 16970 \\ -17479 \\ \hline \end{array}$$

$$509 \text{ cnt} \Rightarrow \underline{2.485 \text{ Ft}}$$

Trolley

$$\begin{array}{r} 15172 \\ 15175 \\ \hline 3 \end{array}$$

$$\Rightarrow \underline{\underline{0.0073 \text{ Ft}}}$$

Waist

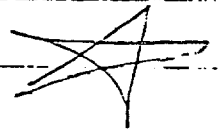
$$\begin{array}{r} 1224 \\ -7136 \\ \hline 88 \end{array}$$

$$\Rightarrow 0.16874 \text{ rad} (0.9668 \text{ deg})$$

Shoulder Pitch

$$\begin{array}{r} 3832 \quad 3455 \\ -4596 \quad -4596 \\ \hline 1074 \quad 1411 \end{array}$$

$$\begin{array}{l} 17.799 \text{ deg} \\ \text{or } 15.5 \text{ deg} \end{array}$$



W Pitch

$$(3987 - 3977) = 10 \Rightarrow 0.44 \text{ deg}$$

W Yaw

$$(4006 - 4003) = 3 \Rightarrow 0.132 \text{ deg}$$

W Roll

$$(5331 - 5324) = 7 \Rightarrow 0.308 \text{ deg}$$

— bridge encoder coupling (hose clamp) needs

to be tightened — possible source of encoder

drifts

— shoulder pitch error is unrealistic, possible

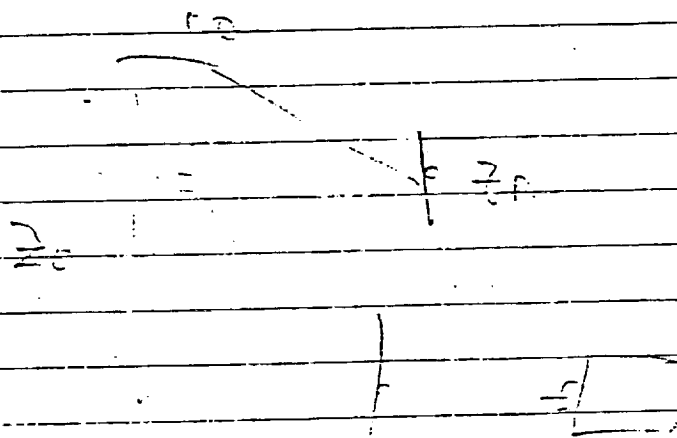
faulty reading from 8/27/96 (do not know if it was

recorded using level) Further investigation needed

9/19/97

# ENCODER ZERO READINGS

|                |                                  |
|----------------|----------------------------------|
| BRIDGE         | 16970                            |
| TROLLEY        | 15172                            |
| WAIST          | 7224                             |
| SHOULDER PITCH | 3822 (SLOPING DOWN) 3485 (LEVEL) |
| ARM EXTENSION  | 256                              |
| WRIST PITCH    | 3977                             |
| WRIST YAW      | 4003                             |
| WRIST ROLL     | 5331                             |



8/27/96 RE-SET ENCODERS

|                |       |
|----------------|-------|
| WRIST YAW      | 4006  |
| WRIST ROLL     | 5324  |
| WRIST PITCH    | 3987  |
| ARM EXTENSION  | 343   |
| SHOULDER PITCH | 4896  |
| WAIST          | 7136  |
| TROLLEY        | 15175 |
| BRIDGE         | 17479 |

OFFSET IS ABOVE VALUE FOR ALL JOINTS  
EXCEPT BRIDGE & TROLLEY

BRIDGE SHOULD READ  $62\frac{7}{16}"$  AT ALIGN MARK

$$(17479 - \text{OFFSET}) \times \text{ENCRES INU} = 62\frac{7}{16}"$$

$$\text{OFFSET} = 17479 - (62\frac{7}{16} / 12) / .0048828 = 17479 - 1065.6 = 16413.4$$

TROLLEY SHOULD READ  $-22\frac{1}{2}"$  AT ALIGN MARK

$$(15175 - \text{OFFSET}) \times \text{ENCRES INU} = -22\frac{1}{2}"$$

$$\text{OFFSET} = 15175 + \frac{22\frac{1}{2}}{12} / .0024414 = 15175 + 768 = 15943$$

- 2) Using the ZOOM utility, acquire the current encoder readings. Adjust AOFFSET such that the encoder positions will be zero for the zero joint positions.

In the joint position equation, AOFFSET compensates the encoder reading so that it will provide a zero value for the zero joint position. For the rotational joints, where the calibration marks can be aligned at the zero angle, the encoder shaft will ideally be at its median position when the marks are aligned (to maximize positive and negative range). This would result in an encoder reading of half the digital range (e.g., 4096 for a 8192 count encoder), the value to which AOFFSET would be set. The arm extension calibration marks align at the joint's retraction limit; the corresponding encoder reading is zero rather than the median reading. The ideal value of AOFFSET would then be zero. The values of AOFFSET for the bridge and trolley are set such that the encoder-derived position will be zero when the waist joint is centered on the B0 origin (remember to account for the offsets that result from the alignment process in step 1 when setting these values). Table 1 provides the ranges and resolutions of the encoders that were installed in 9/93.

Table 1 - Encoder Constants

| <u>Joint</u> | <u>Encoder<br/>Range<br/>(counts)</u> | <u>Encoder<br/>Resolution<br/>(ft./cnt. or rad./cnt.)</u> |
|--------------|---------------------------------------|---|
| Bridge       | 32,768                                | 0.0048828   |
| Trolley      | 32,768                                | 0.0024414   |
| Waist        | 8192                                  | 0.00019175  |
| Shoulder     | 8192                                  | 0.00019175  |
| Arm Ext.     | 8192                                  | 0.0012207   |
| W Pitch      | 8192                                  | 0.00076699  |
| W Yaw        | 8192                                  | 0.00076699  |
| W Roll       | 8192                                  | 0.00076699  |

## Reference 8

DCI TM#011398-1  
2BODY Code Modifications and Test Runs  
January 13, 1998



## **Dynamic Concepts Technical Memorandum #011398-1**

To: bd Systems - Mr. Ronald Francis

From: Dynamic Concepts - Dr. Patrick Tobbe, bd Systems - Mr. Mike Ekbundit

Subject: 2BODY Code Modifications and Test Runs

Date: January 13, 1998

### **1.0 Introduction**

The real time simulation 2BODY, which is hosted on the SGI Challenge machine UQBAR and simulates two rigid bodies in contact, was modified to facilitate its usage, streamline its code, and speed up its run time. A simplified temporary main routine was built to run test cases independent of the 2BODY real time executive routine developed for parallel operation. Given acceptable results, the 2BODY software can then be integrated with the new parallel executive routines and processes. The twelve cases test uniaxial force and torque loads about the three cardinal axes. This memorandum will describe the software modifications made to 2BODY and the test cases.

### **1.1 Modifications**

The 2BODY code was ported to UQBAR from the Alliant computer ALLI as a separate program from the latest version of ROCKET. All of the include files based in the 2BODY code were combined to form a new structure, TB. The CTL(control) and DYN(dynamics) routines in the COMP and DOCK directories were then combined to form new routines that would control both the COMP and DOCK processes. The COMP source code directories were then deleted and twelve test cases were run on the new code.

## **2.0 SOFTWARE MODIFICATIONS**

### **2.1 Structure Addition**

All of the include files based in the 2BODY code with the prefix "cmmn" were combined to form a new structure named "s\_tb.inc". The five include files that were combined are shown in Table 1. "cmmnsix.inc" is the only file that was not included in the new structure because it is based in the SIXDOF directory. The prefix "TB" was added to all of the variables in the new structure and the structure name is included in the argument lists of the necessary CALL, ENTRY, and SUBROUTINE statements.

|              |
|--------------|
| s_tb.inc     |
| cmmn.inc     |
| cmmn1.inc    |
| cmmn2.inc    |
| cmmnbb.inc   |
| cmmncomp.inc |

TABLE 1 - Include files combined into structure TB

## 2.2.1 Combination of COMP and DOCK files

The COMP and DOCK programs were combined to form a single executable that controls both functions. The COMP and DOCK programs were made up of routines with similar filenames so the code in the COMP files that was dissimilar to the code in the DOCK files was cut and pasted into the DOCK files. "start.f", "fmtran.f", and "comp.f" were created from "startdk.f", "startcomp.f", "fmtrandk.f", "fmtranc.f", "compc.f", and "compdk.f". The new file structure can be seen in Figure 1 in the appendix. Appropriate logic was added to these files to run either compensation or docking tests.

## 2.2.2 Compensation Coordinate Frames

The coordinate frames for the compensation tests are set up as shown in figures 3 through 7 in the appendix. These coordinate frames are for the compensation testing on the CBM, common berthing mechanism, only. "compin.dat", the data file used in compensation testing, is based on these coordinate frames. The user is responsible for changing the coordinate frames in the case of a hardware change. Also, the single axis loads in the COMP tests always act along the Z axis.

"testin.dat", the data file used for docking tests, defines a coordinate frame transformation matrix D1N. The transformation matrix D2M is currently hard coded for the CBM also and must be changed in "start.f" in the case of a hardware change.

## 2.2.3 Execution Instructions

2BODY is run by moving to uqbar:usr/people/tobbe/di/exec/2body and typing "2body". This starts the 2BODY code after which the user will be asked a series of questions relevant to the process being run. The user will first be prompted to choose a hardware or software only run, then prompted to choose COMP or DOCK. COMP and DOCK runs use the input data files "compin.dat" and "testin.dat" respectively. The difference between the data files is in the coordinate frame definitions. Only an output file name is required when executing a DOCK run. During a COMP run, the user will be prompted to enter

the compensation source for the run. The compensation value can be read from a curve in the data file or from the keyboard. The compensation coefficient, AOT for translation and BOT for rotation, is then entered if the compensation value is from the keyboard. Finally, the user will be required to input an output file name for the COMP run. Figure 2 in the appendix illustrates the execution of 2BODY.

### 3.0 Example Runs

Twelve test runs, consisting of six compensation runs and six docking runs, were designed to test the basic functionality of the new 2BODY code. The docking tests include uniaxial forces and torques, applied one at a time, about each of the three cardinal axes. The compensation tests include runs with compensation values of zero, two, and four, for a uniaxial force and torque about the local Z axis.

#### 3.1 Run Descriptions

Table 2 illustrates the twelve test cases for 2BODY. Test cases one through six are docking runs and cases seven through twelve are compensation runs. The moment of inertia and mass of body one are  $10^{35}$  lb-ft-sec<sup>2</sup> and  $10^{25}$  slugs respectively and the moment of inertia and mass of body two are 10 lb-ft-sec<sup>2</sup> and 10 slugs respectively to simulate an infinite mass body making contact with a small body.

| TEST CASE | COMP/DOCK | COMP VALUE | LOAD TYPE | LOAD AXIS | LOAD     |
|-----------|-----------|------------|-----------|-----------|----------|
| 1         | DOCK      | NONE       | FORCE     | X         | 10lb     |
| 2         | DOCK      | NONE       | FORCE     | Y         | 10lb     |
| 3         | DOCK      | NONE       | FORCE     | Z         | 10lb     |
| 4         | DOCK      | NONE       | TORQUE    | X         | 10 ft-lb |
| 5         | DOCK      | NONE       | TORQUE    | Y         | 10 ft-lb |
| 6         | DOCK      | NONE       | TORQUE    | X         | 10 ft-lb |
| 7         | COMP      | 0          | FORCE     | Z         | 10lb     |
| 8         | COMP      | 0          | TORQUE    | Z         | 10 ft-lb |
| 9         | COMP      | 2          | FORCE     | Z         | 10lb     |
| 10        | COMP      | 2          | TORQUE    | Z         | 10 ft-lb |
| 11        | COMP      | 4          | FORCE     | Z         | 10lb     |
| 12        | COMP      | 4          | TORQUE    | Z         | 10 ft-lb |

TABLE 2 - 2BODY test cases

### 3.2 Prediction of Test Results

The single axis tests were designed to make the results easily verifiable by using mass, inertia, and load values which simplify the equations of motion. If one assumes that body one is stationary, the motion of body two is the relative motion between the two bodies. The single axis translational and angular accelerations of body two,  $a_2$  and  $\alpha_2$ , are given by

$$F_2 = m_2 a_2$$

$$T_2 = I_2 \alpha_2$$

Therefore the position of body two can be found by

$$R_2 = v_0 t + \frac{1}{2} a_2 t^2$$

for constant force loads where the initial velocity is zero. The corresponding velocity of body two when  $v_0$  is zero is

$$\dot{R}_2 = a_2 t$$

Similarly, because the applied torques are uniaxial, angular position can be found by

$$\theta_2 = \omega_0 t + \frac{1}{2} \alpha_2 t^2$$

When  $\omega_0$  is zero, the angular velocity can be found by

$$\dot{\theta}_2 = \alpha_2 t$$

The test cases were set up such that  $F_2 = 10$  lb,  $I_2 = 10$  lb-ft-sec<sup>2</sup>,  $T_2 = 10$  ft-lb and  $m_2 = 10$  ~~lb~~<sup>sl</sup>. This forces  $a_2$  and  $\alpha_2$  to be 1 ft/s<sup>2</sup> and 1 rad/s<sup>2</sup> respectively and leads to

$$\theta_2 = R_2 = \frac{1}{2} t^2$$

and

$$\dot{\theta}_2 = \dot{R}_2 = t$$

where  $R_2$  and  $\dot{R}_2$  have units of ft and ft/s, respectively, and  $\theta_2$  and  $\dot{\theta}_2$  have units of rad and rad/s, respectively. Also, because body one is an infinite mass body and remains stationary through contact with body two, the relative motion between body one and body two is the same as the motion of body two.

Graphs of the position and velocity of body two during docking runs should be a half parabola and a line respectively, according to the previous equations. In fact, graphs of docking runs where a uniaxial force was applied show position starting at zero and increasing parabolically to 50 ft at 10 seconds. Also, the velocity of body two for the same cases also starts at zero and increases linearly to 10 ft/s at 10 seconds. For cases of uniaxial applied torque, the Euler angles for body two start at zero and increase parabolically to 2rad at 2 seconds and the angular velocity  $\omega$  starts at zero and increases linearly to 10 rad/s at 10 seconds. The graphs of the docking runs are in the appendix.

An intrinsic part of compensation testing is the compensation factor, the factor used by the simulation to minimize the lag between a 6-DOF table command and table response. This is incorporated into the relative motion equations as

$$R_{D2D1} = R_{D2D1} + AOT \bullet \dot{R}_{D2D1}$$

where AOT is the translational compensation factor and

$$\theta_{D2D1} = \theta_{D2D1} + BOT \bullet \dot{\theta}_{D2D1}$$

where BOT is the rotational compensation factor. Both AOT and BOT are scalar values. However, because the uncompensated relative motion between bodies one and two is the same as the motion of body two

$$R_{D2D1} = R_2 = \frac{1}{2}at^2$$

$$\theta_{D2D1} = \theta_2 = \frac{1}{2}\alpha t^2$$

The equations for relative motion can be rewritten as

$$R_{D2D1} = \frac{1}{2}at^2 + AOT \bullet \dot{R}_{D2D1}$$

and

$$\theta_{D2D1} = \frac{1}{2}\alpha t^2 + BOT \bullet \dot{\theta}_{D2D1}$$

Table 3 shows the results of compensation testing. It should be noted that when the compensation value is zero, the compensation and docking runs will yield the same results.

| COMP VAL | LOAD     | AXIS | OUTPUT TERM | TIME(sec) | VALUE  |
|----------|----------|------|-------------|-----------|--------|
| AOT=0    | 10lb     | Z    | RD2D1D1     | 10        | 50 ft  |
| BOT=0    | 10 ft-lb | Z    | THETAZ      | 2         | 2 rad  |
| AOT=2    | 10lb     | Z    | RD2D1D1     | 10        | 70 ft  |
| BOT=2    | 10 ft-lb | Z    | THETAZ      | 10        | 70 rad |
| AOT=4    | 10lb     | Z    | RD2D1D1     | 10        | 90 ft  |
| BOT=4    | 10 ft-lb | Z    | THETAZ      | 10        | 90 rad |

Table 3 - Compensation test results

#### 4.0 Conclusions

2BODY has been successfully ported to UQBAR. A new structure, "TB", has been added to the 2BODY code, creating a new shared memory region for the 2BODY variables. The COMP and DOCK programs were successfully integrated into one new executable that controls both the COMP and DOCK simulations. All twelve test cases were completed successfully and the test results were corroborated through hand calculations. The next step is to integrate 2BODY with the real time parallel executive routine.

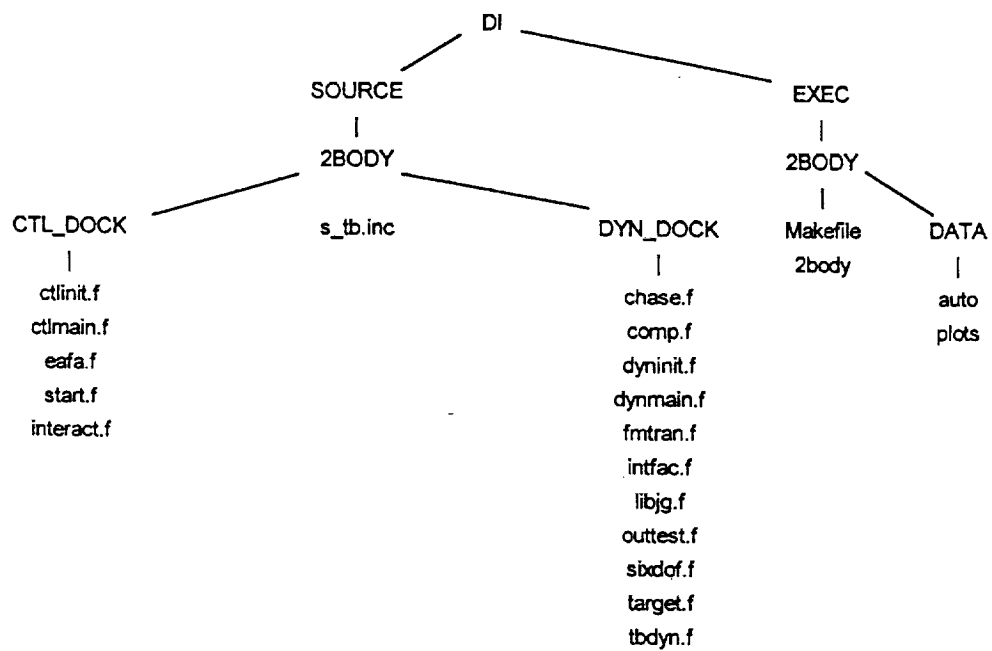


FIGURE 1 - Updated file organization diagram

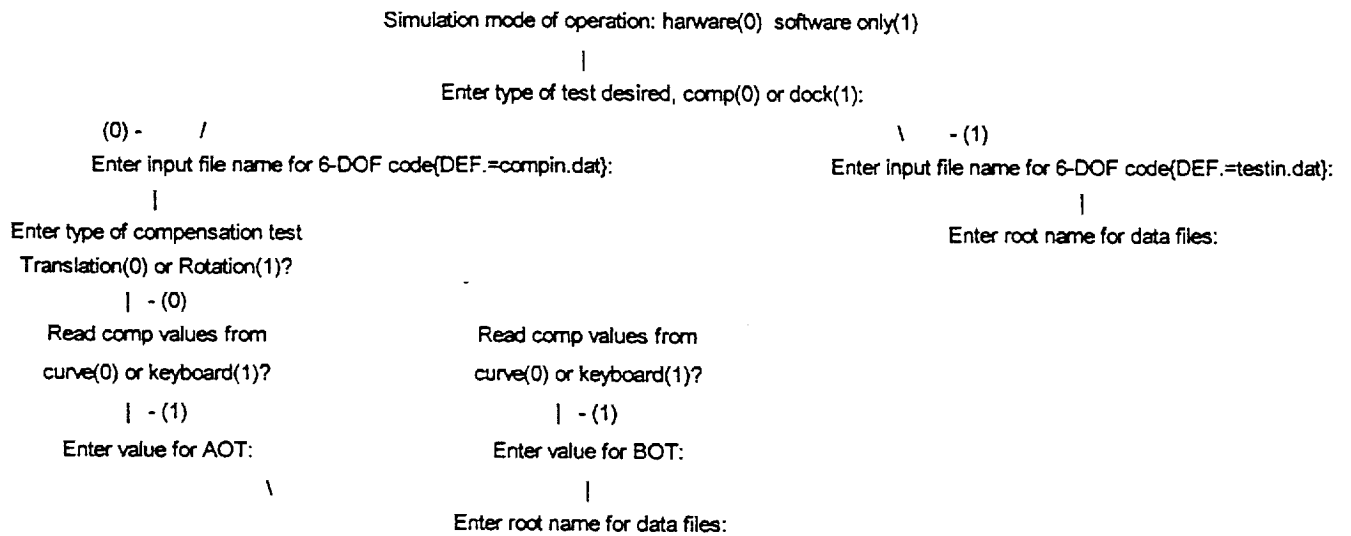


Figure 2 - 2BODY execution diagram



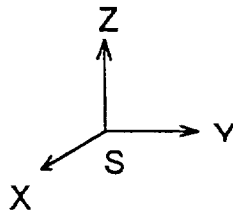


Figure 3 - S coordinate frame

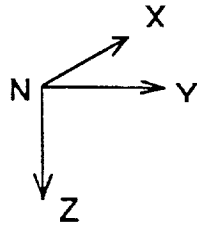


Figure 4 - N coordinate frame

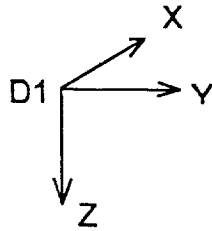


Figure 5 - D1 coordinate frame

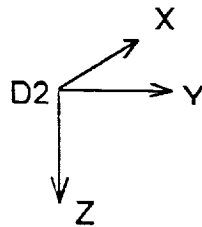


Figure 6 - D2 coordinate frame

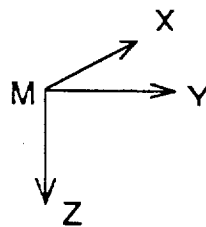


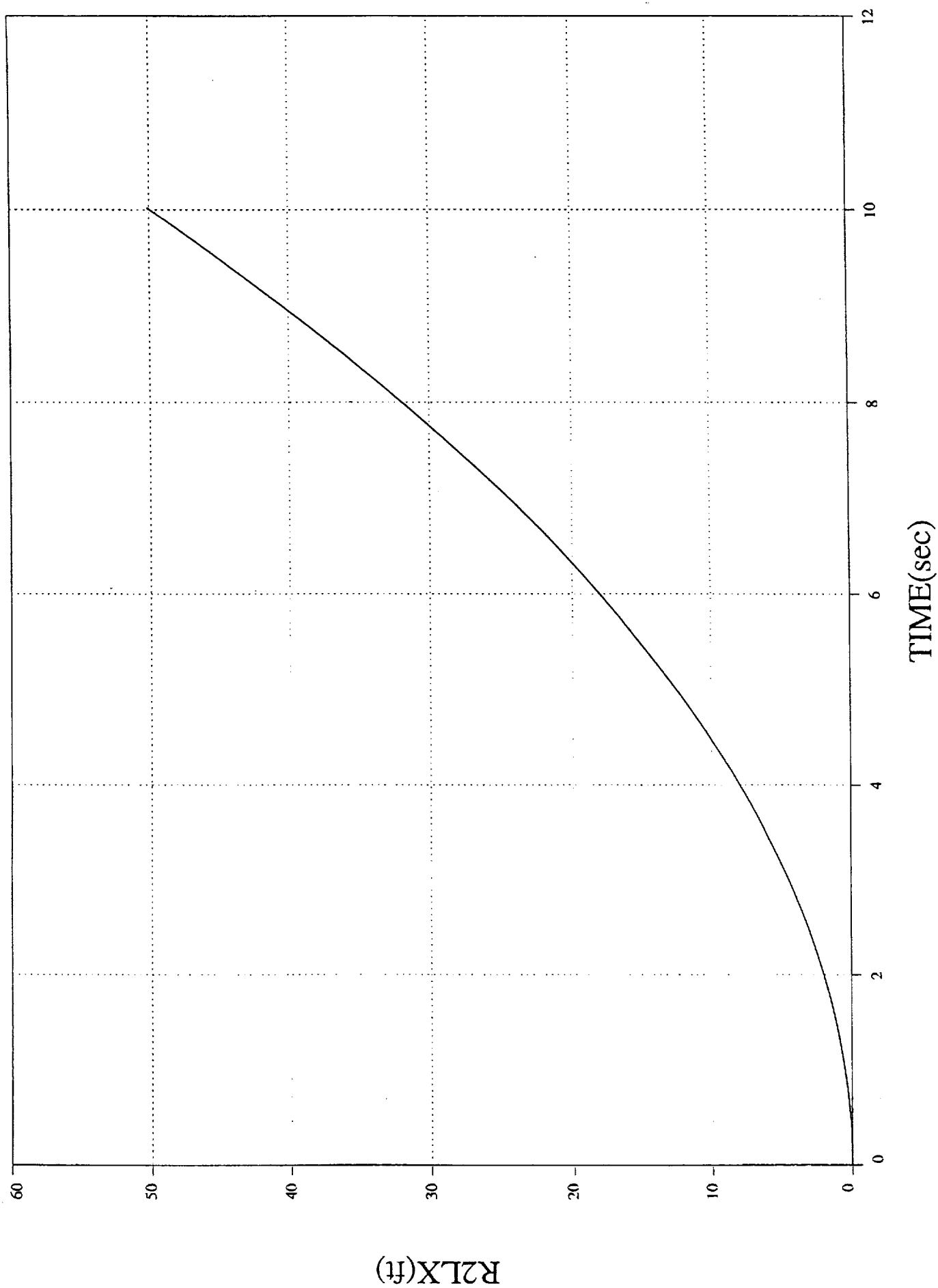
Figure 7 - M coordinate frame

## Appendix

TEST 1

# DI/2BODY Position Body 2 vs. Time

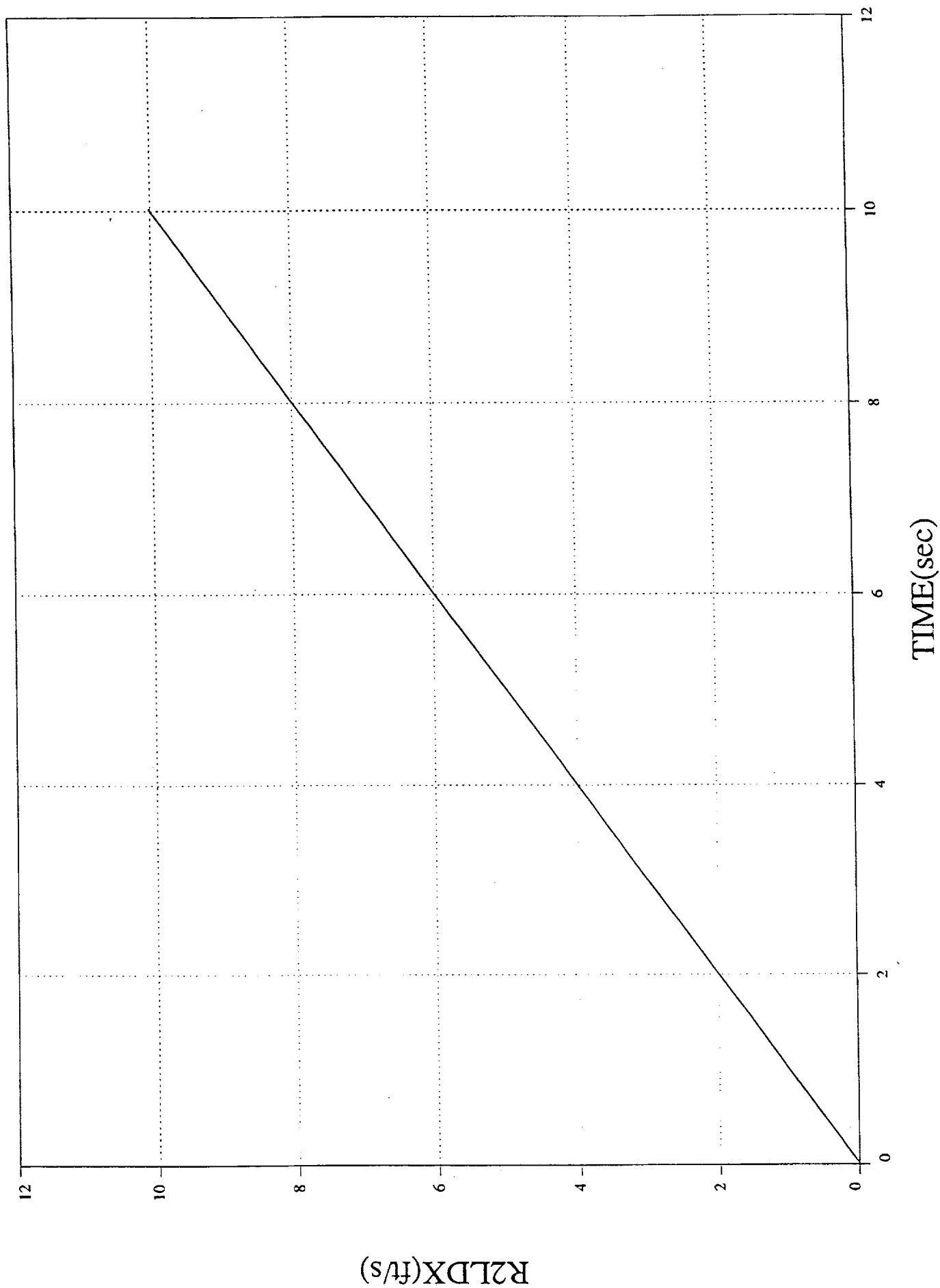
2bdf1  
1 0 25



TEST1

# DI/2BODY Velocity Body 2 vs. Time

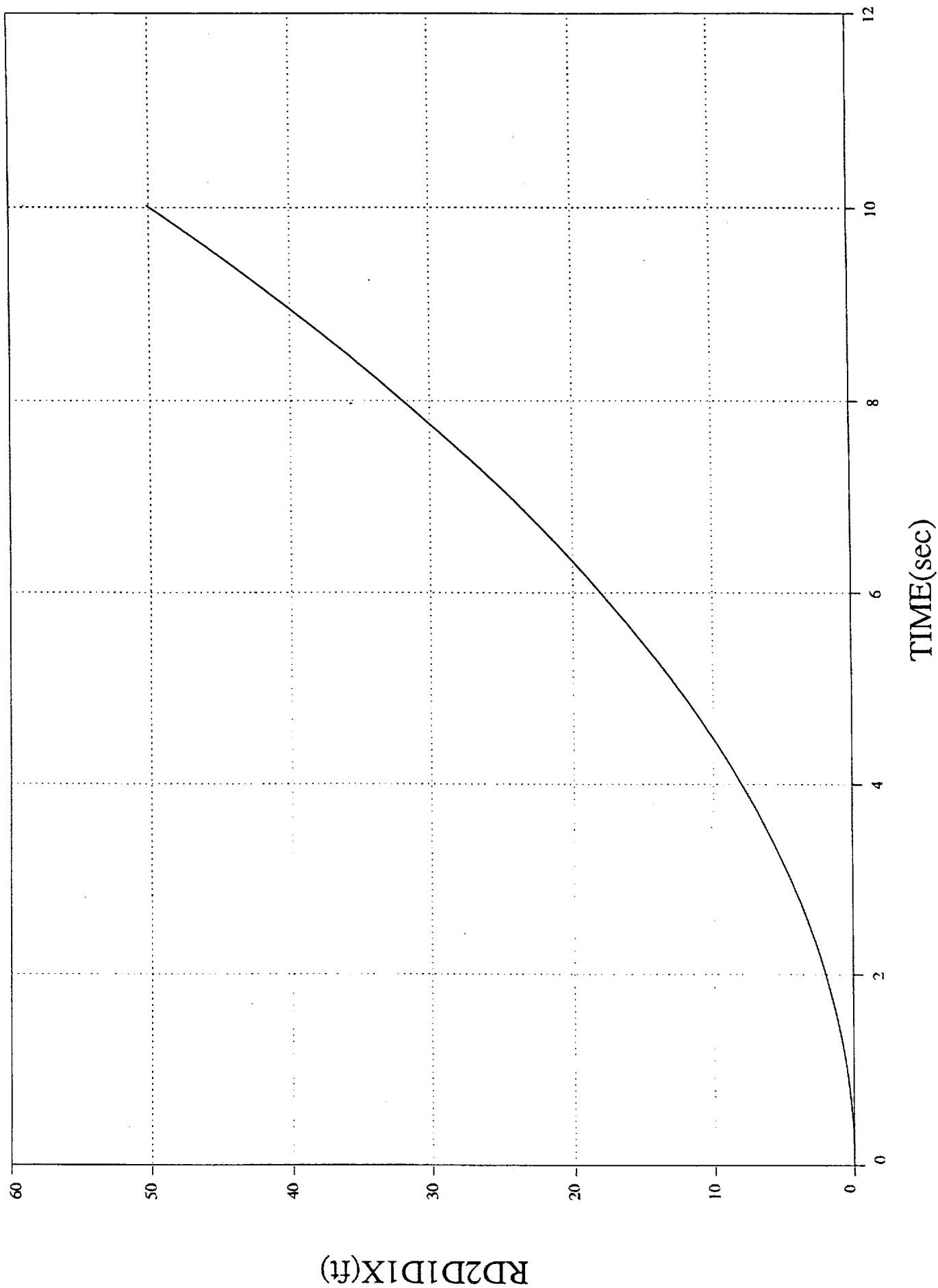
2bdf1  
1 0 — 22



TEST 1

DI/2BODY Rel Pos Body1-2 vs. Time

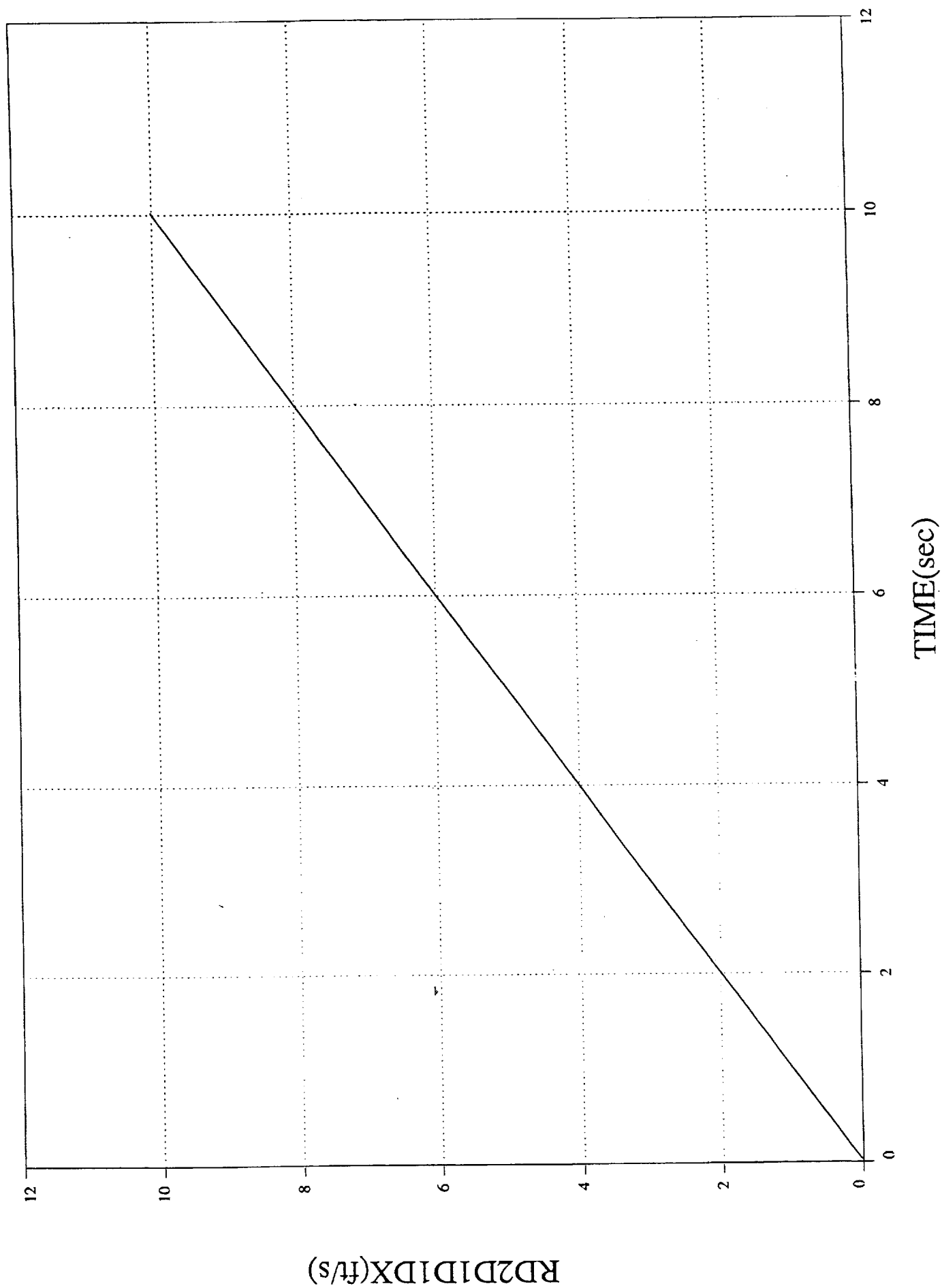
2bddf1  
1 0 — 0 2



TEST 1

DI/2BODY Rel Vel Body1-2 vs. Time

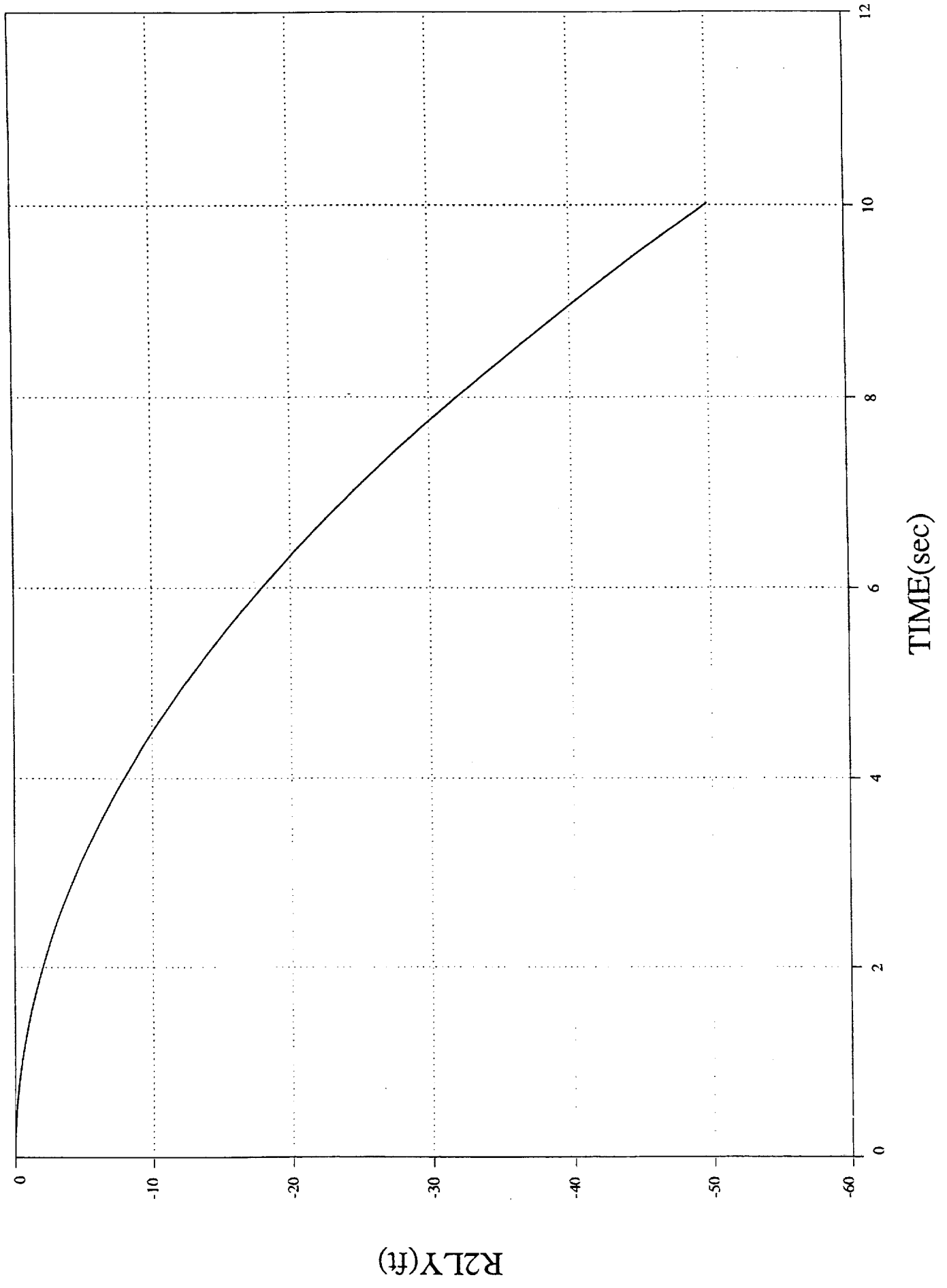
2bddf1  
1 ○ — ○ 28



TEST 2

# DI/2BODY Position Body 2 vs. Time

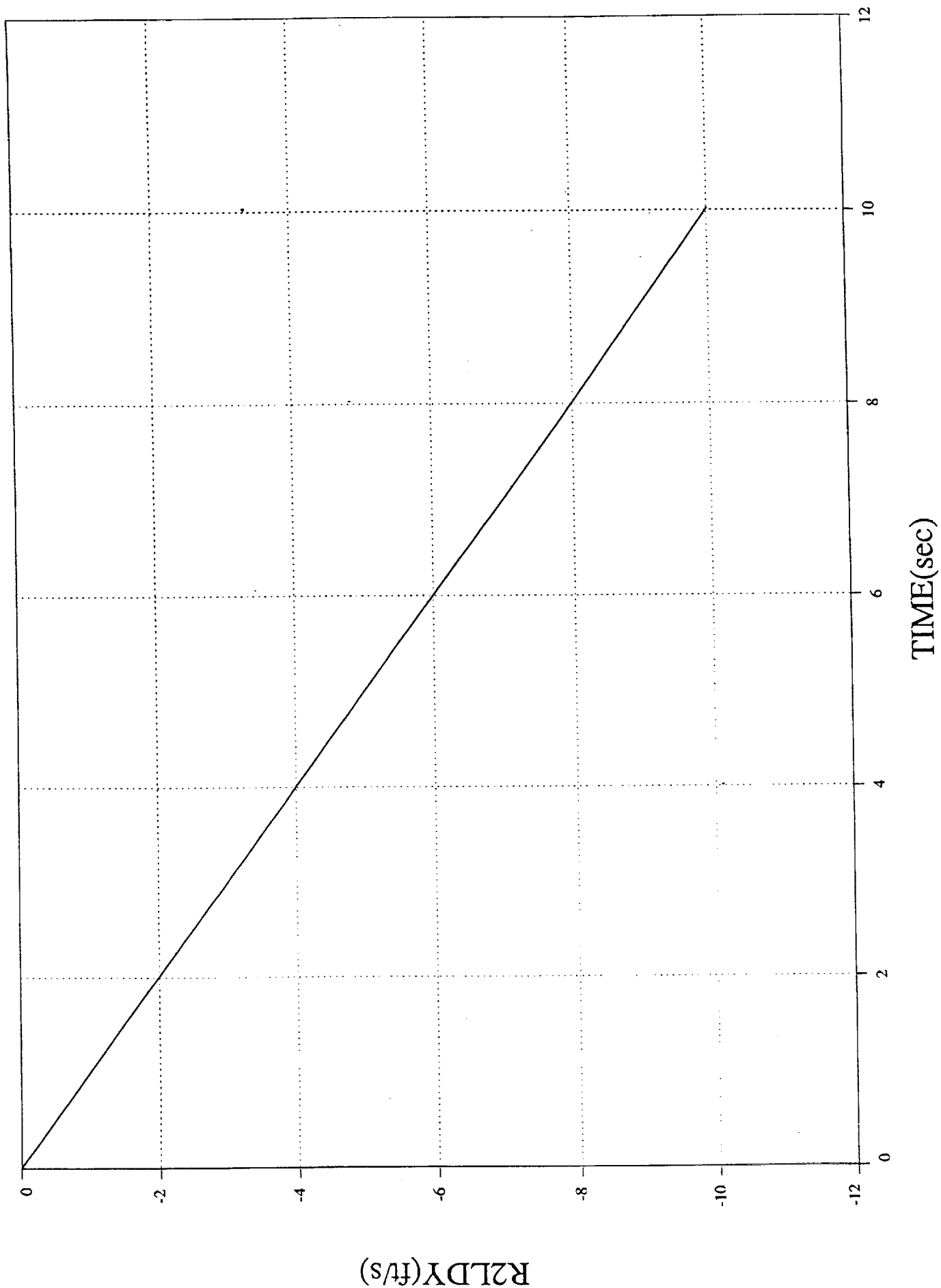
2bddf2  
1 0 — 0 26



TEST 2

# DI/2BODY Velocity Body 2 vs. Time

2bddf2  
1 0 — 23

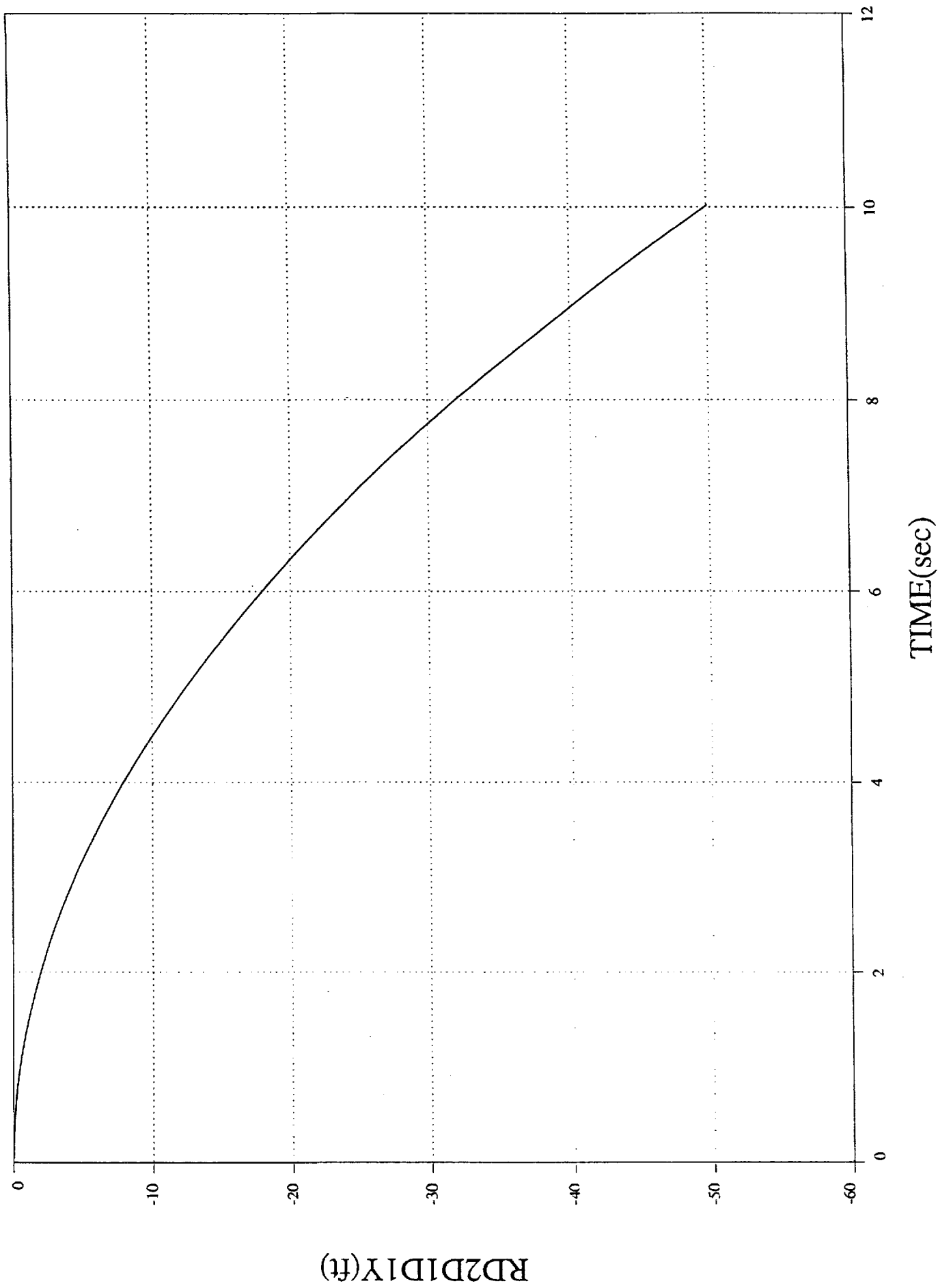




TEST 2

DI/2BODY Rel Pos Body1-2 vs. Time

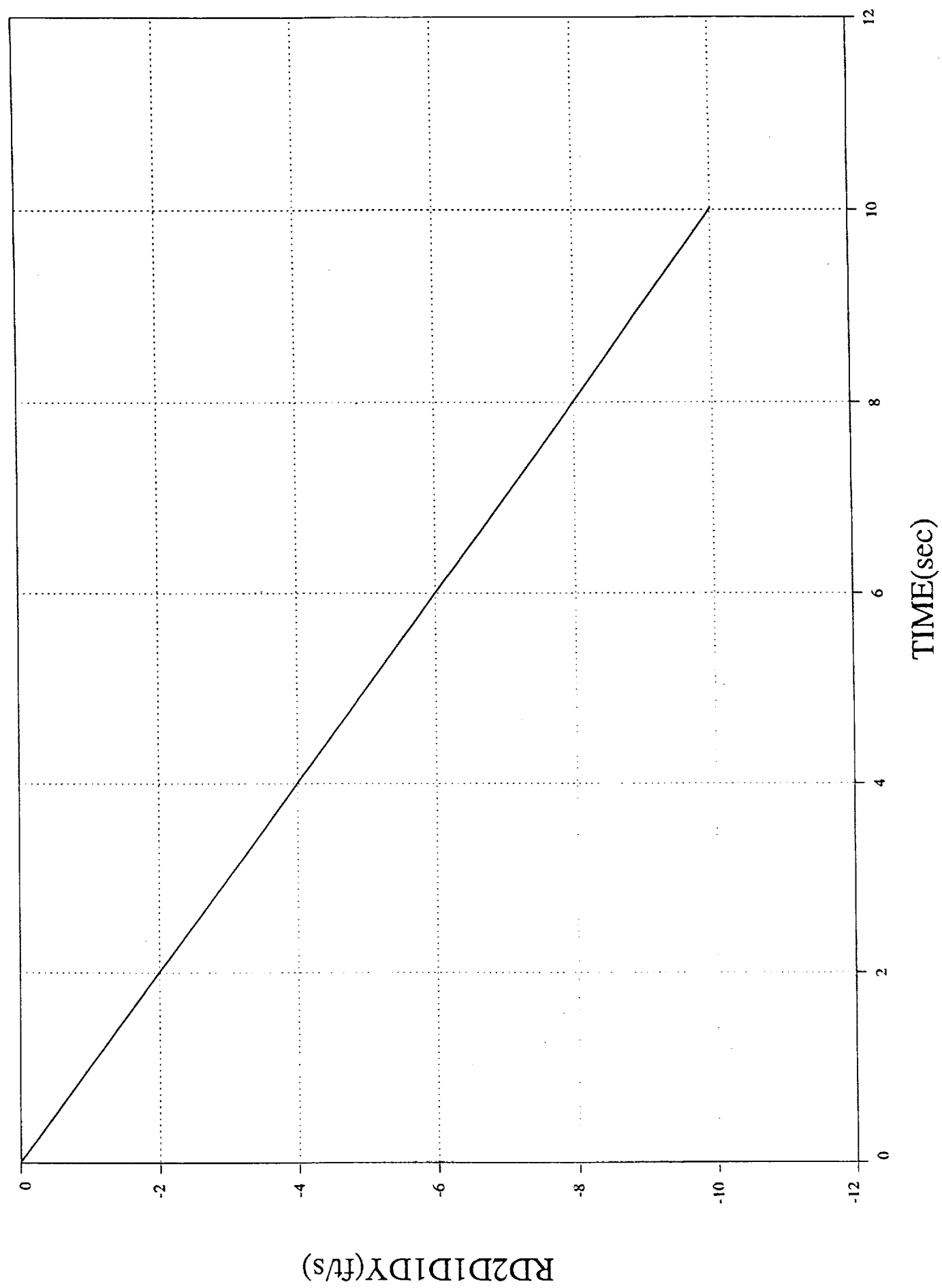
2bd1f2  
1 0 3



1 (ES) 2

# DI/2BODY Rel Vel Body1-2 vs. Time

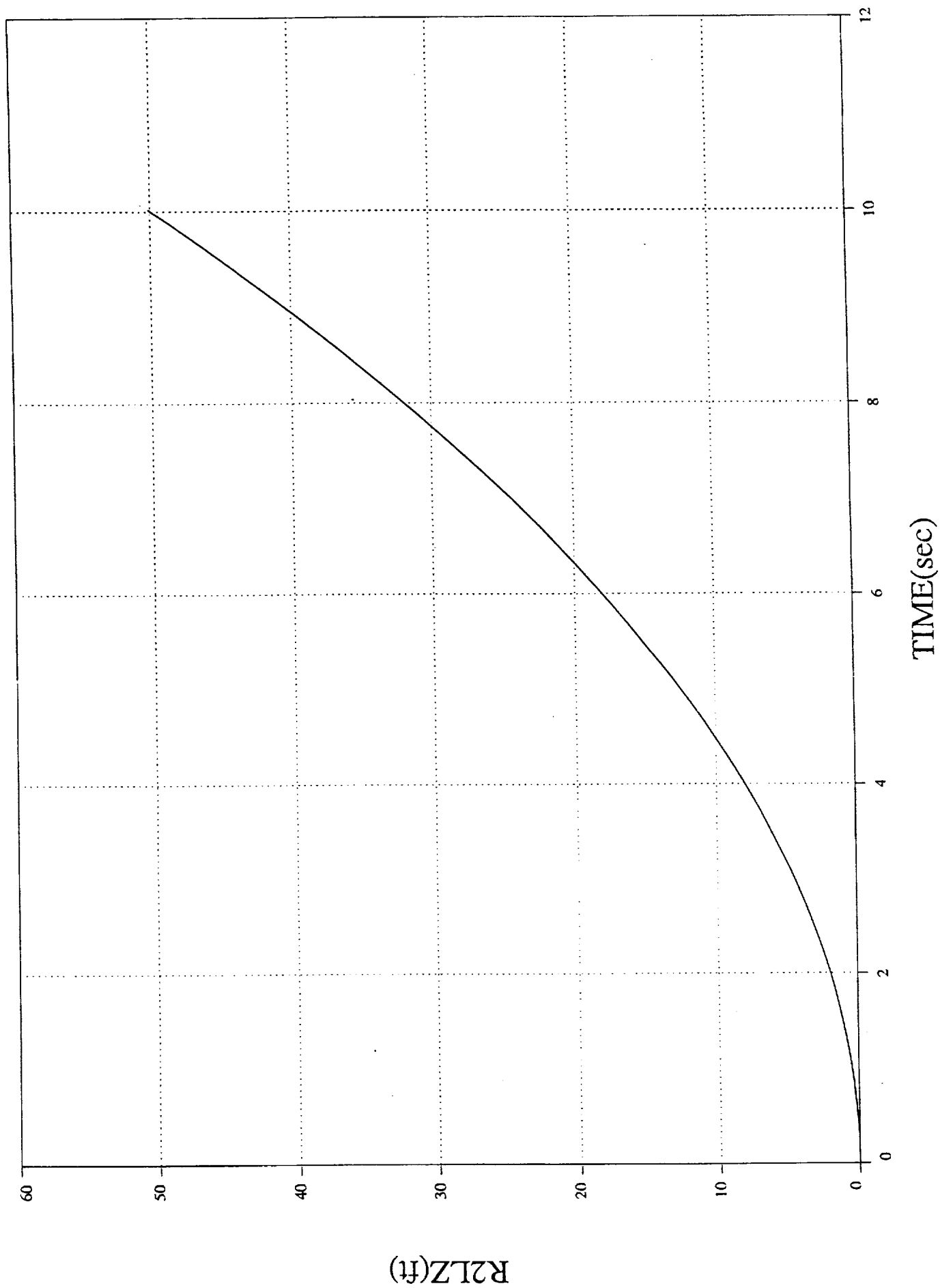
2bddf2  
1 0 — 29



TEST 3

# DI/2BODY Position Body 2 vs. Time

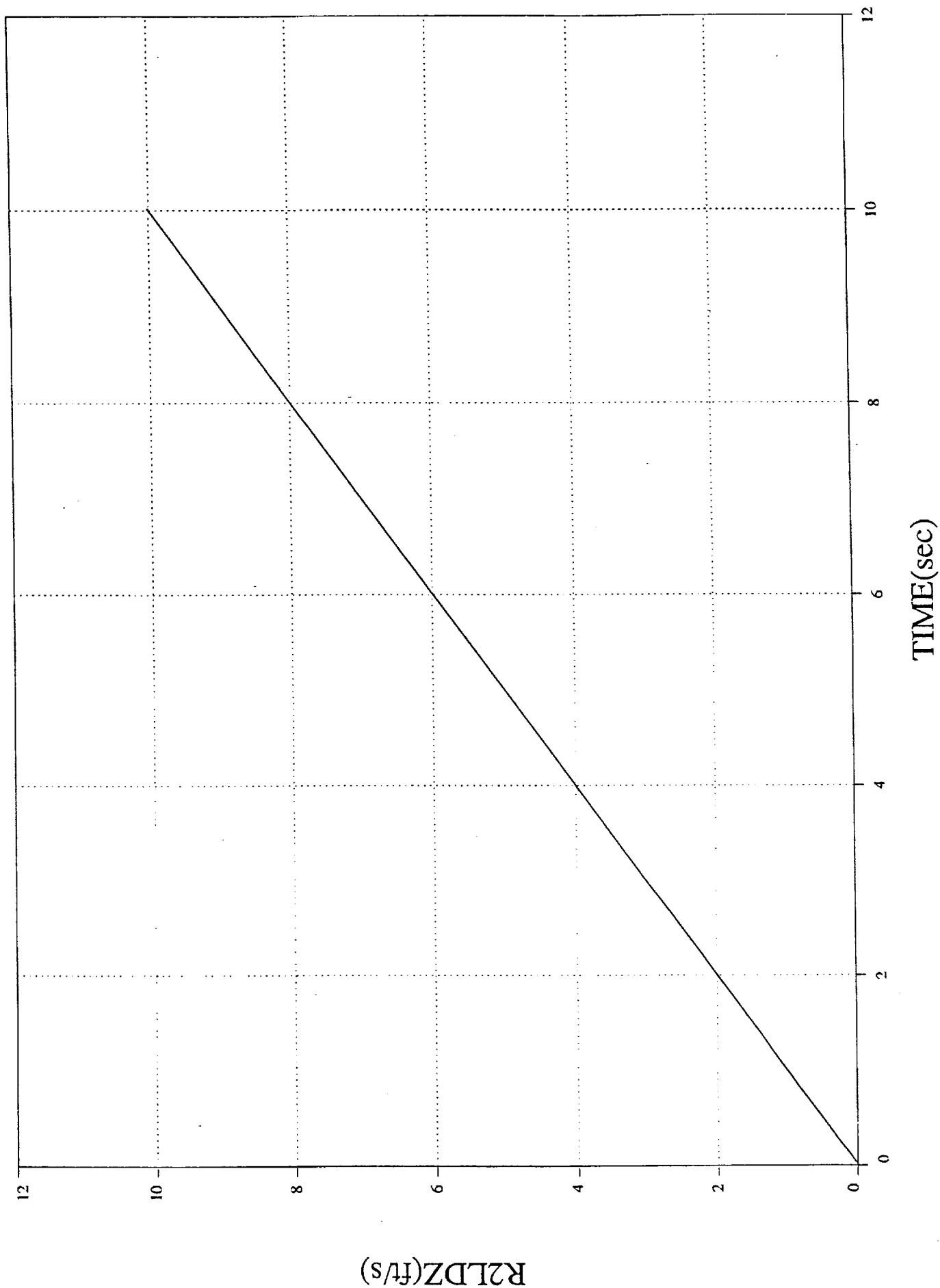
2bddf3  
1 0 — 27



TEST 3

# DI/2BODY Velocity Body 2 vs. Time

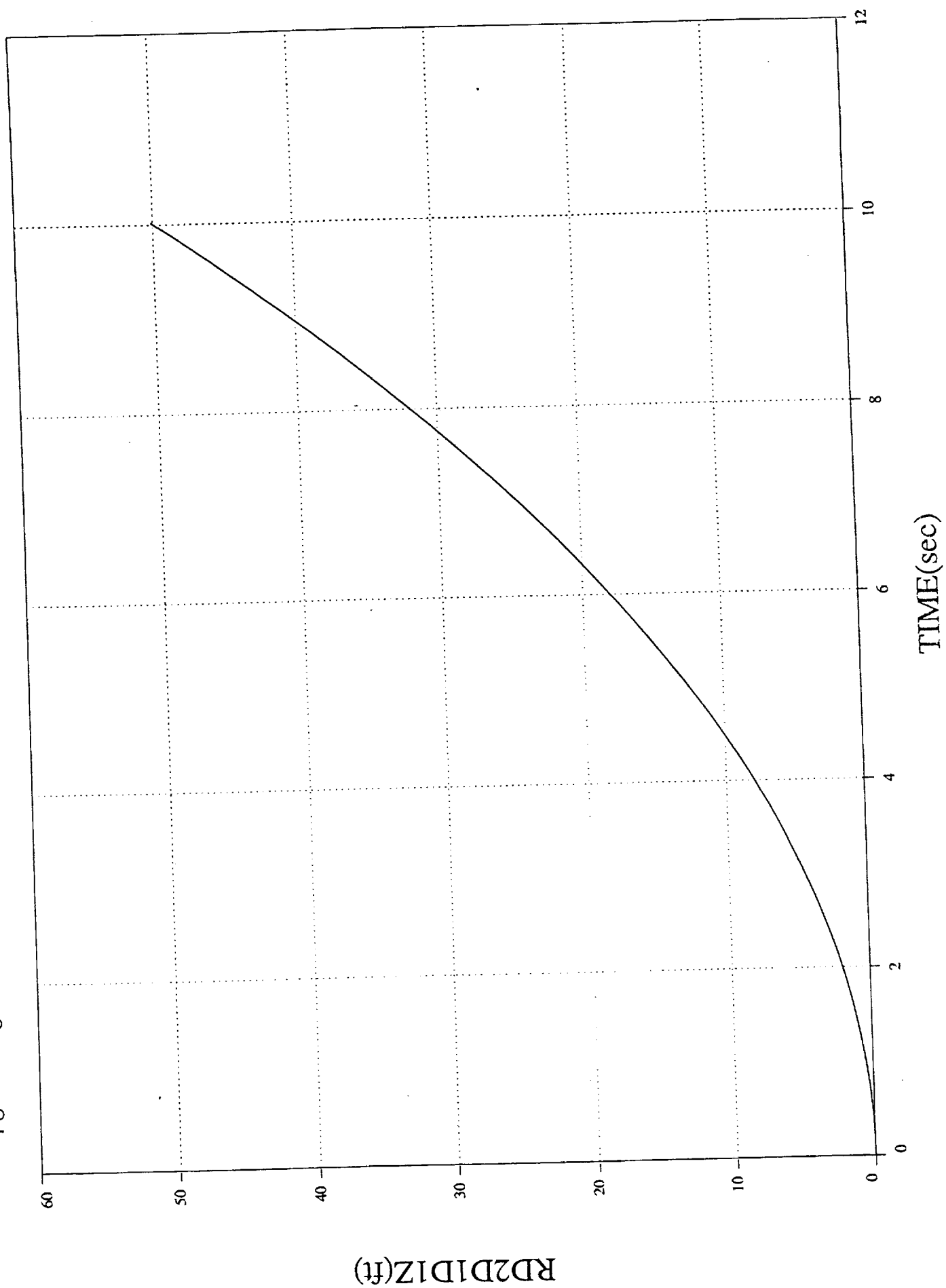
2bddf3  
1 G — 24



TEST 3

DI/2BODY Rel Pos Body 1-2 vs. Time

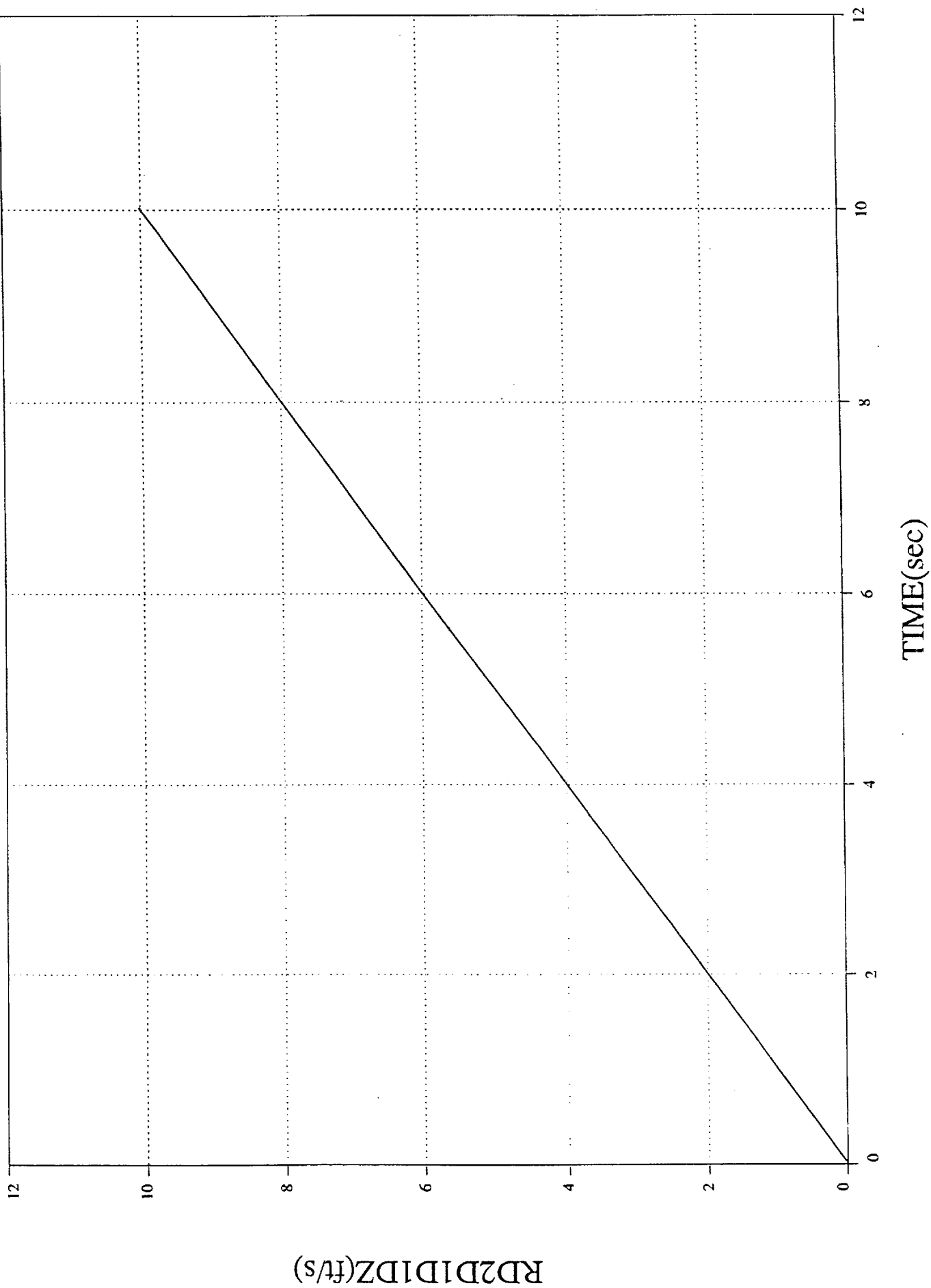
2bddf3  
1 0 — 4



TEST 3

# DI/2BODY Rel Vel Body1-2 vs. Time

2bdf3  
1 0 — 30

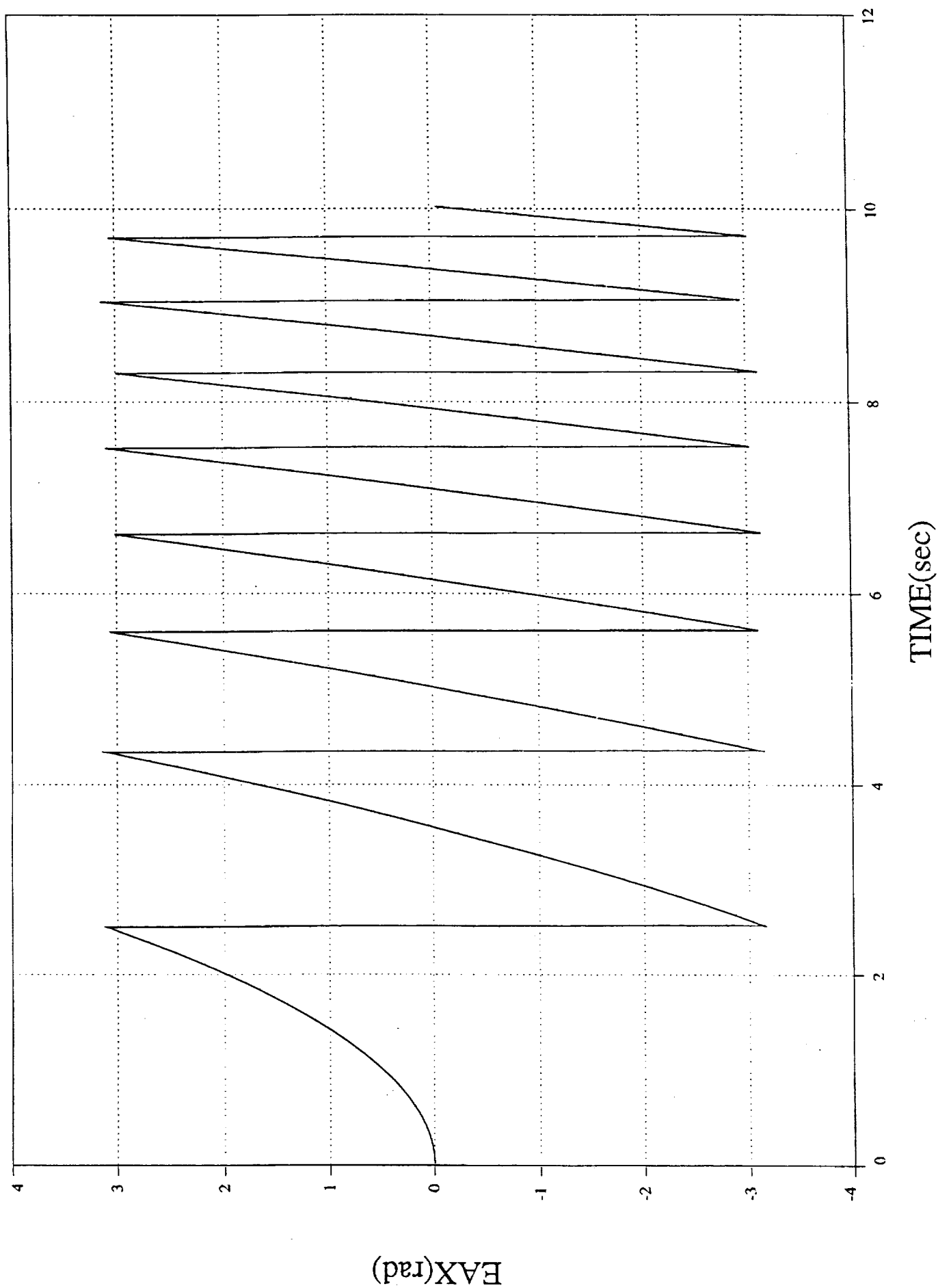


TEST 4

# DI/2BODY Euler Angle 2 vs. Time

2bdt11

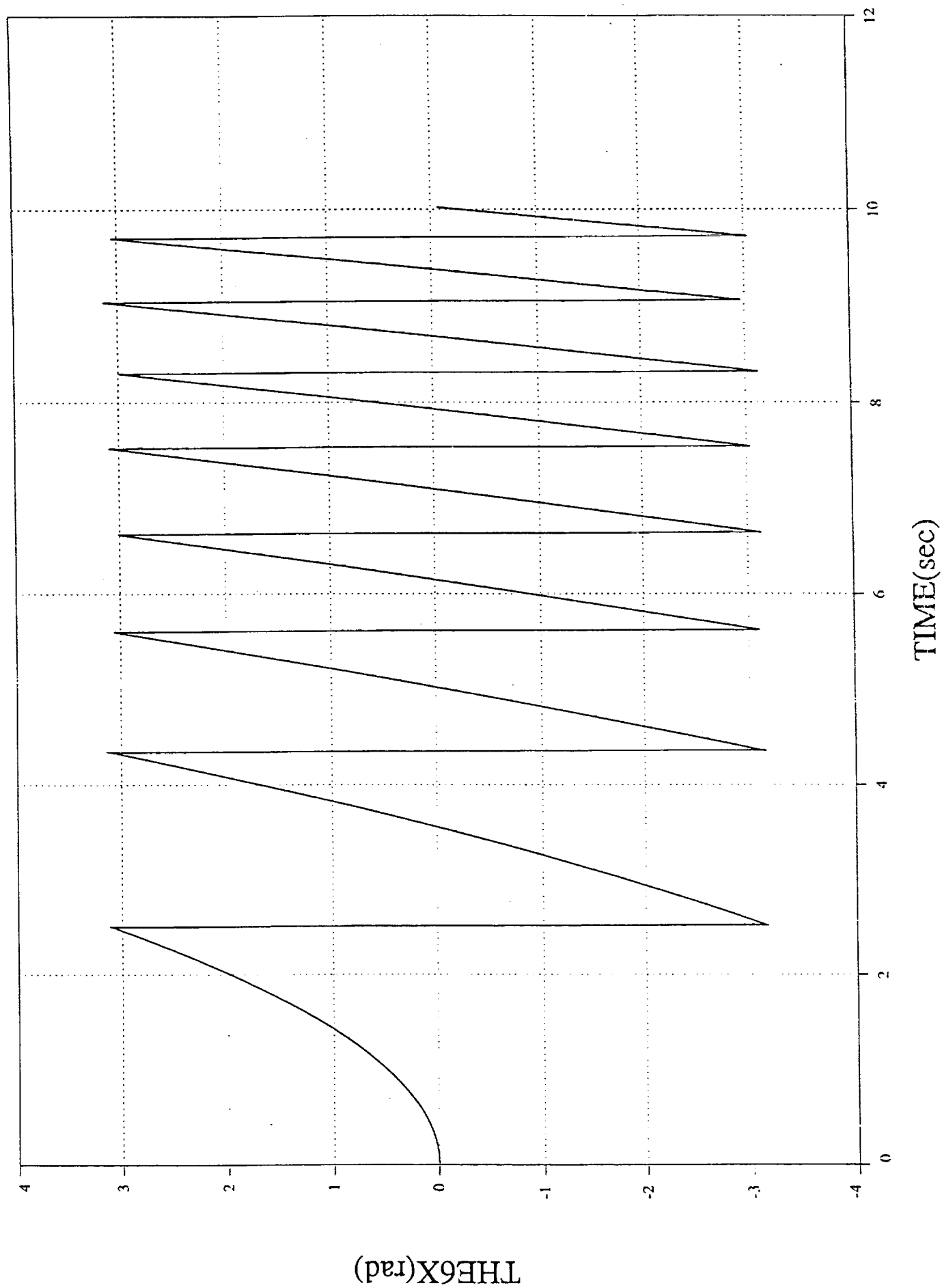
1 0 5



TEST 4

# DI/2BODY Rel Rotation vs. Time

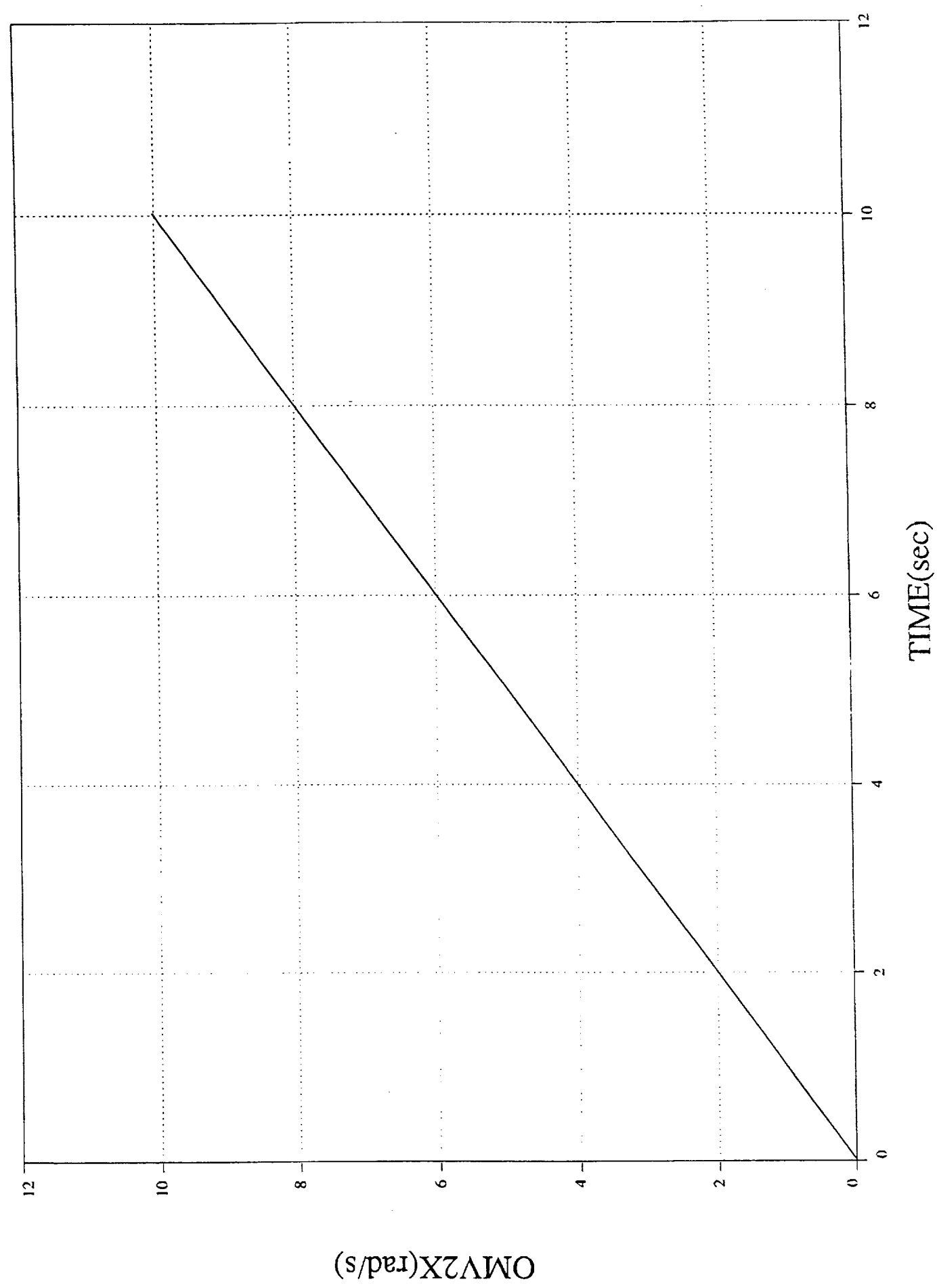
2bddd1  
1 0 37





# DI/2BODY Angular Velocity vs. Time

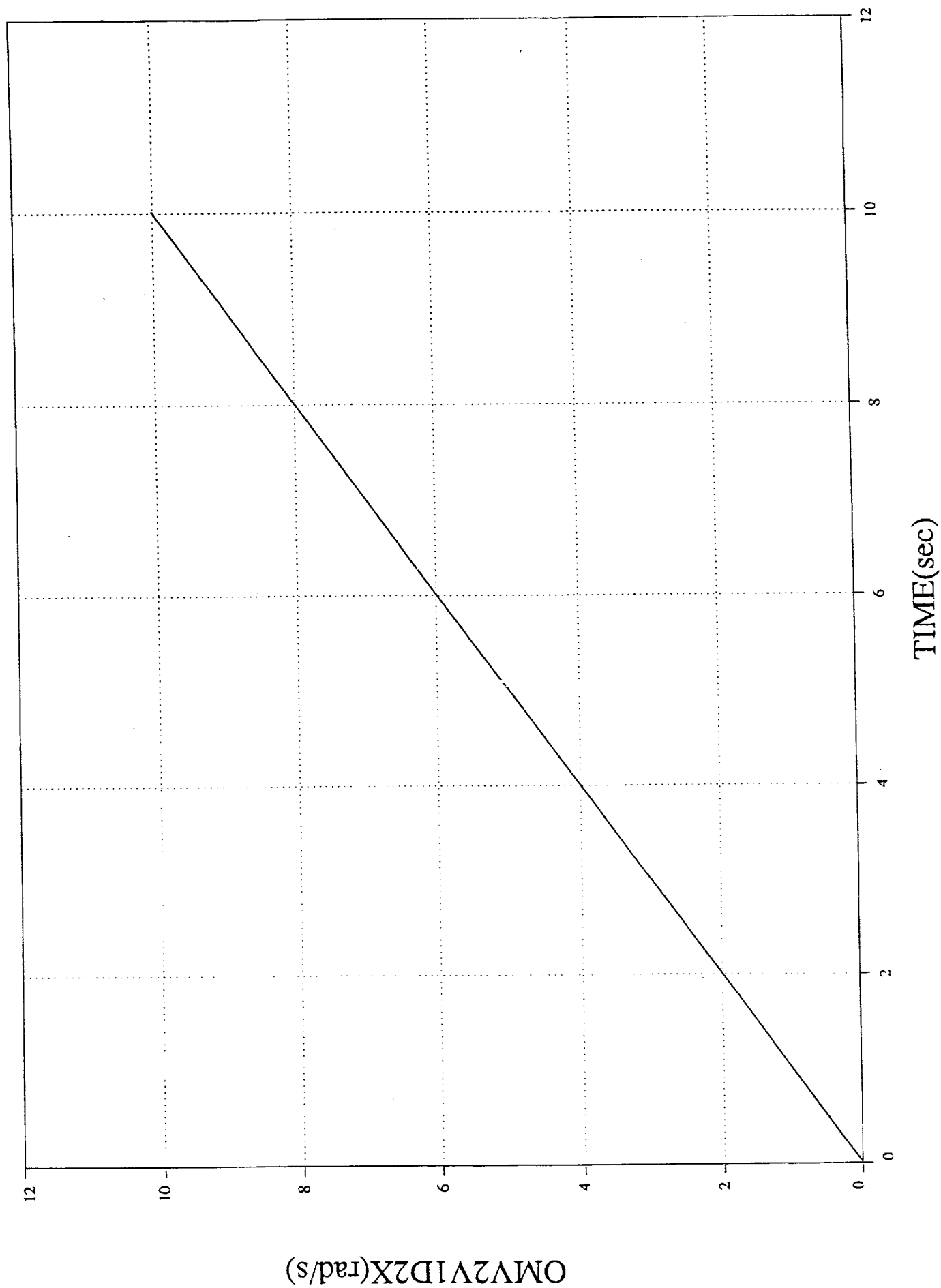
2bdt1  
1 0 31



TEST 4

# DI/2BODY Rel Ang Velocity vs. Time

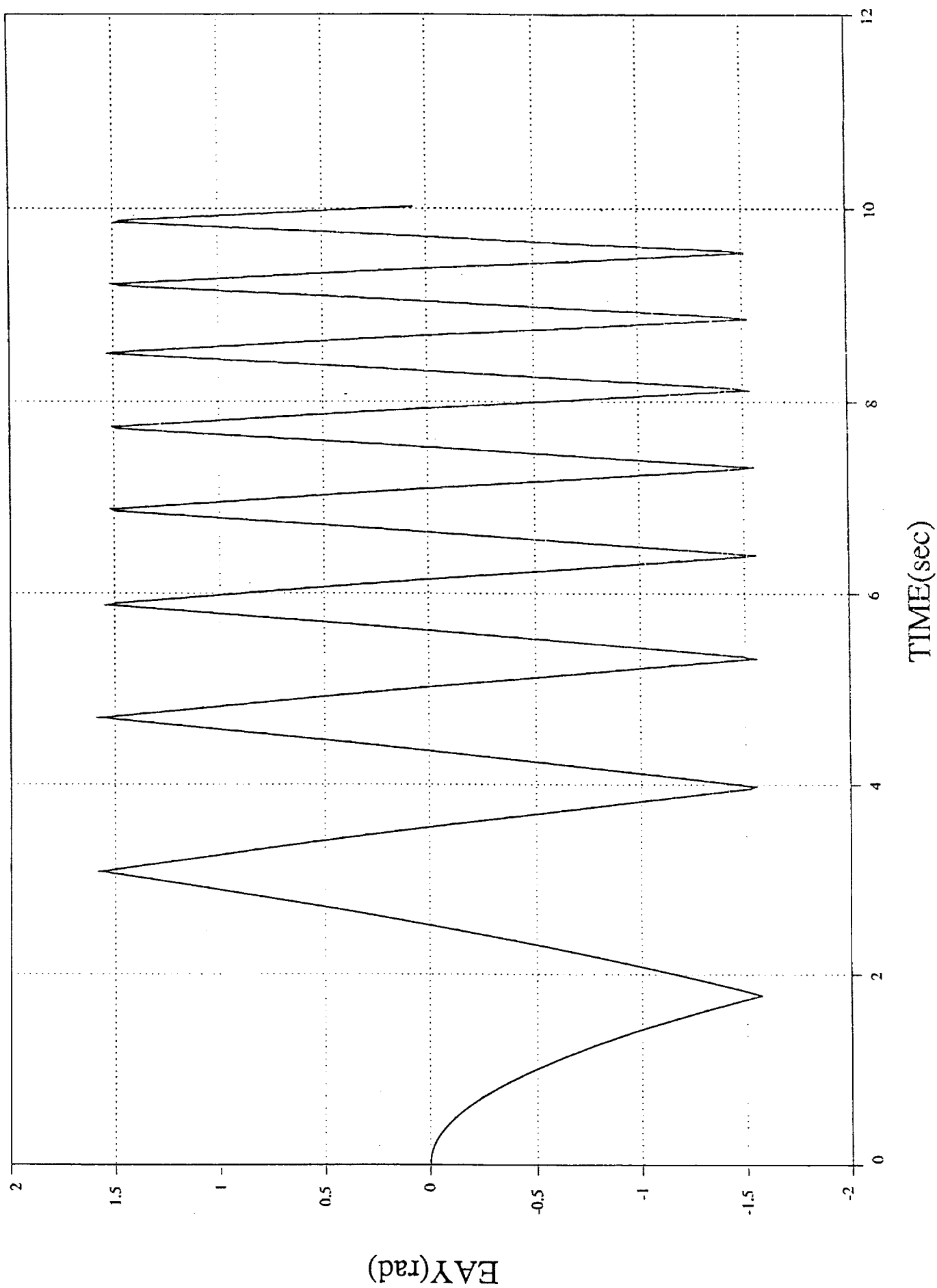
2bdt1  
1 0 — 8



TEST 5

# DI/2BODY Euler Angle 2 vs. Time

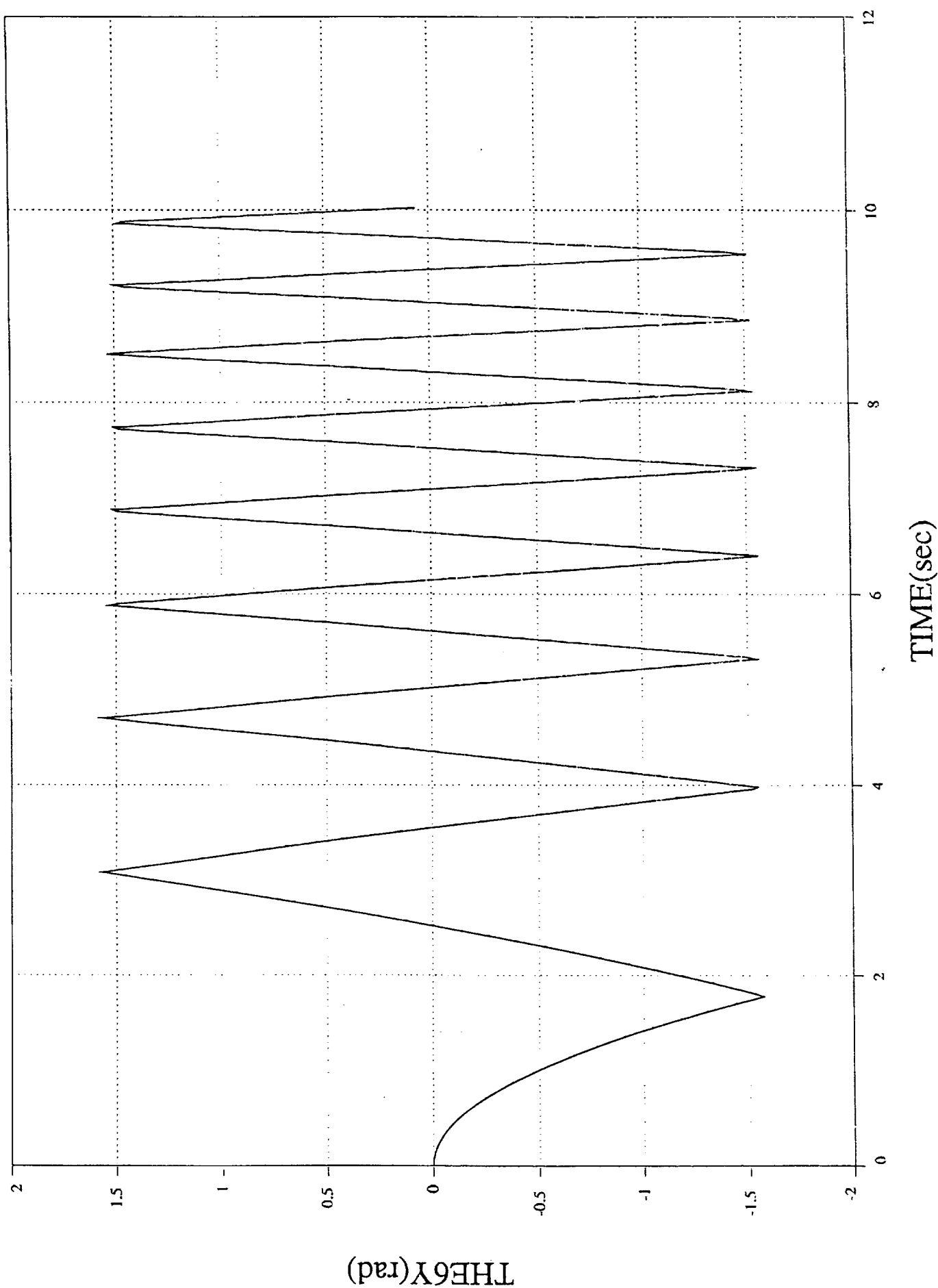
2bddd2  
1 0 6



TEST 5

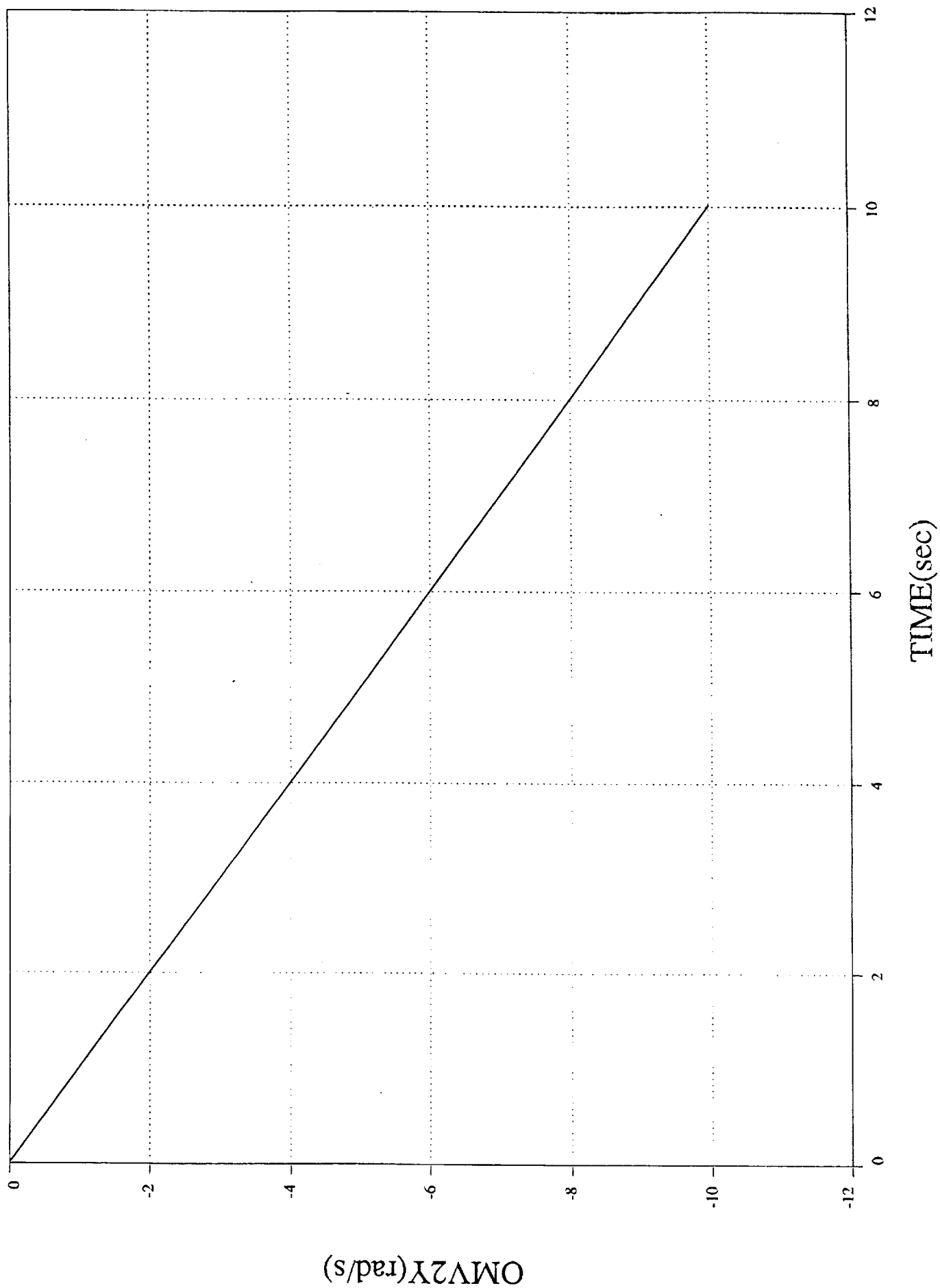
# DI/2BODY Rel Rotation vs. Time

2bdat2  
1 0 — 38



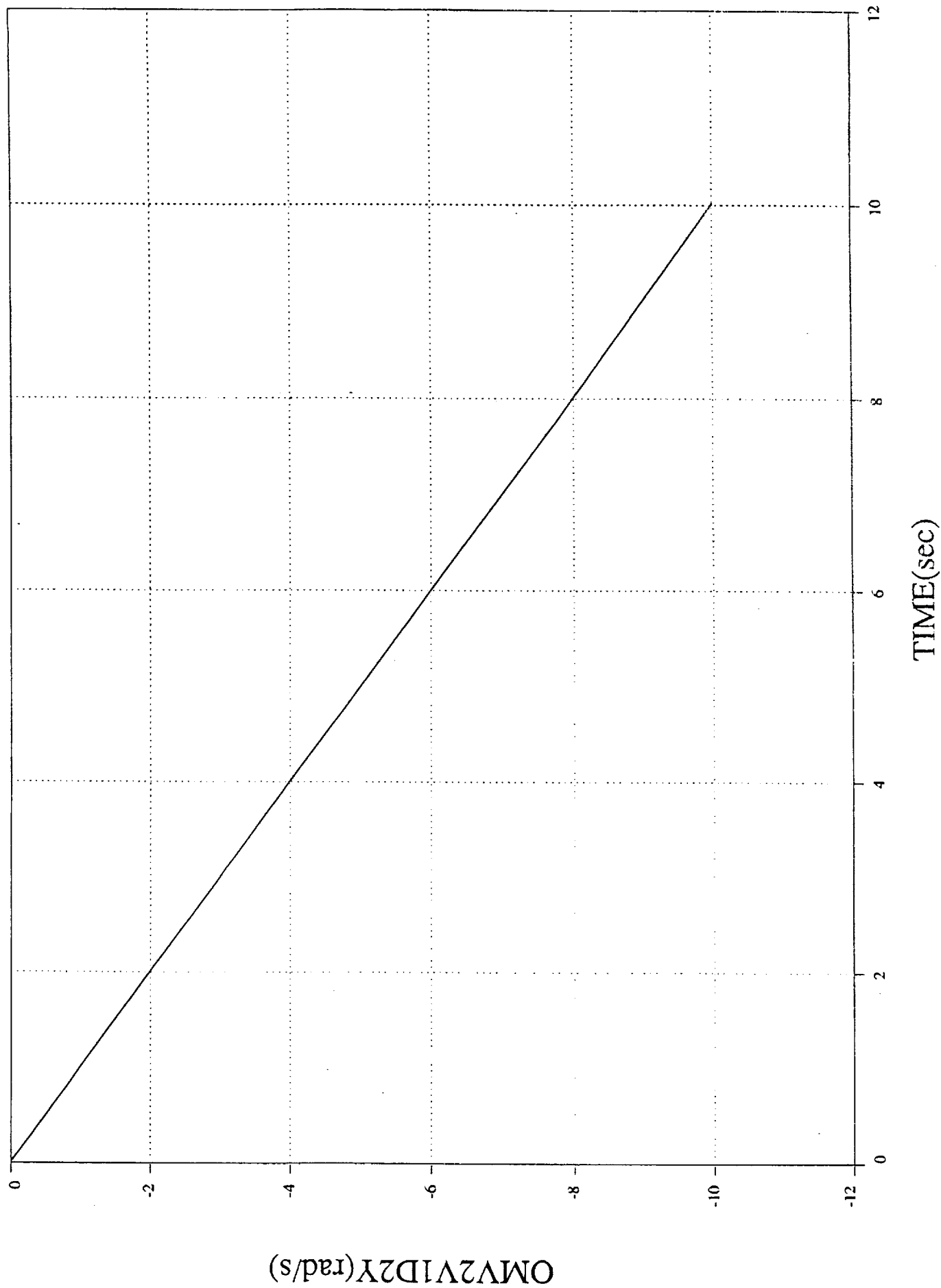
# DI/2BODY Angular Velocity vs. Time

2bdt2  
1 0 32



# DI/2BODY Rel Ang Velocity vs. Time

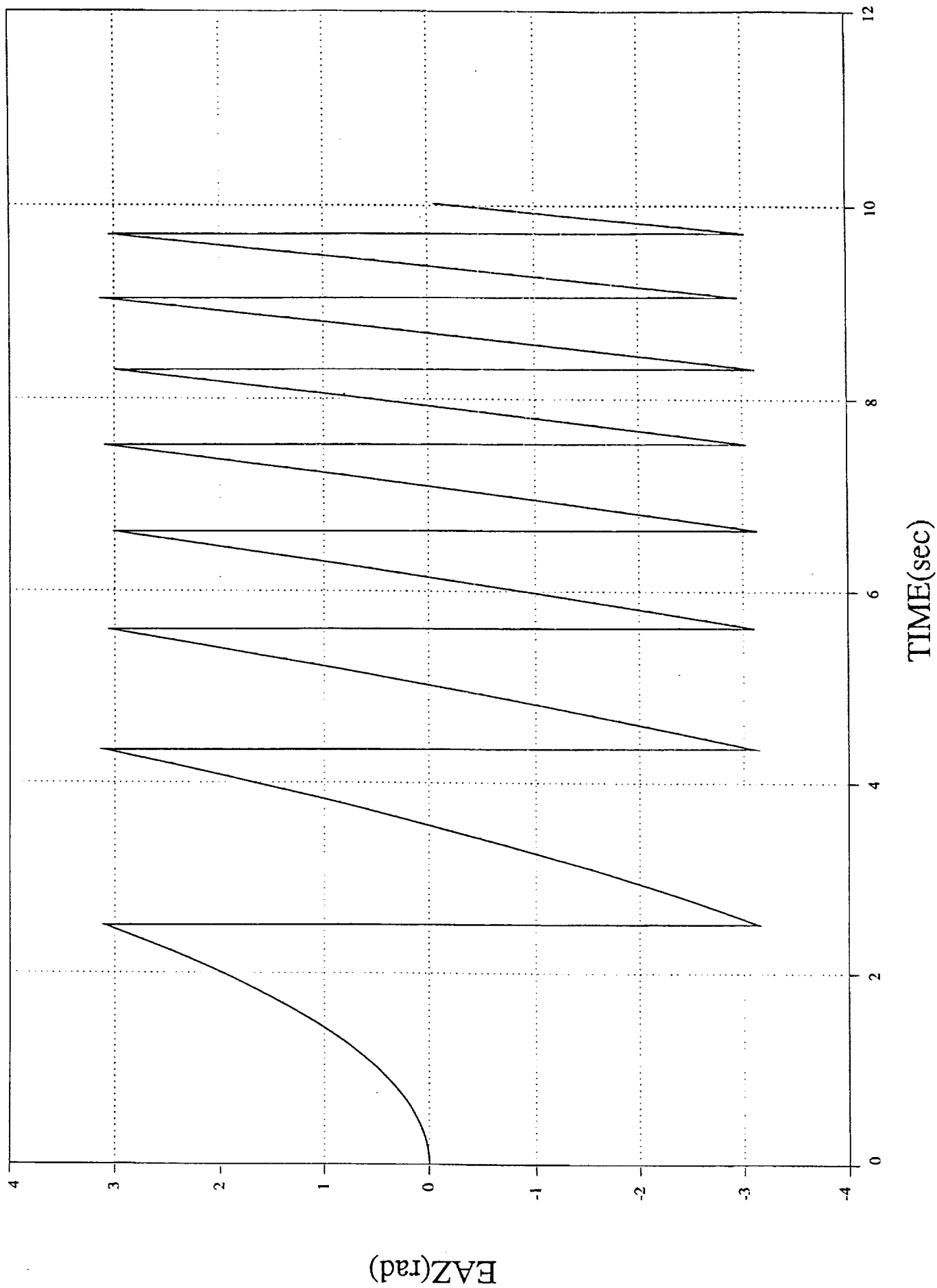
2bddd2  
1 0 — 0 9



TEST 6

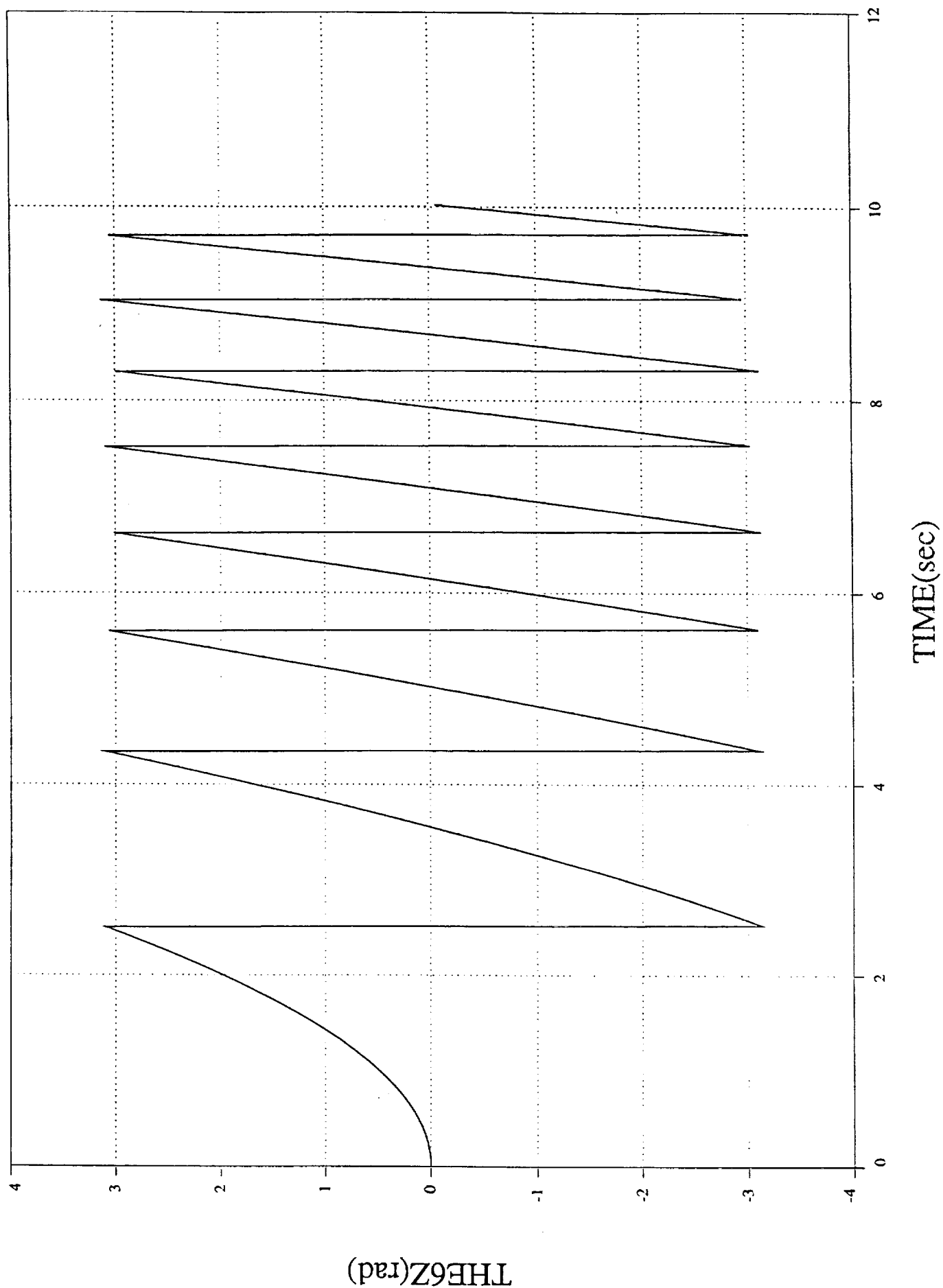
# DI/2BODY Euler Angle 2 vs. Time

2bdt3  
1 0 — 5



# DI/2BODY Rel Rotation vs. Time

2bdat3  
1 0 39

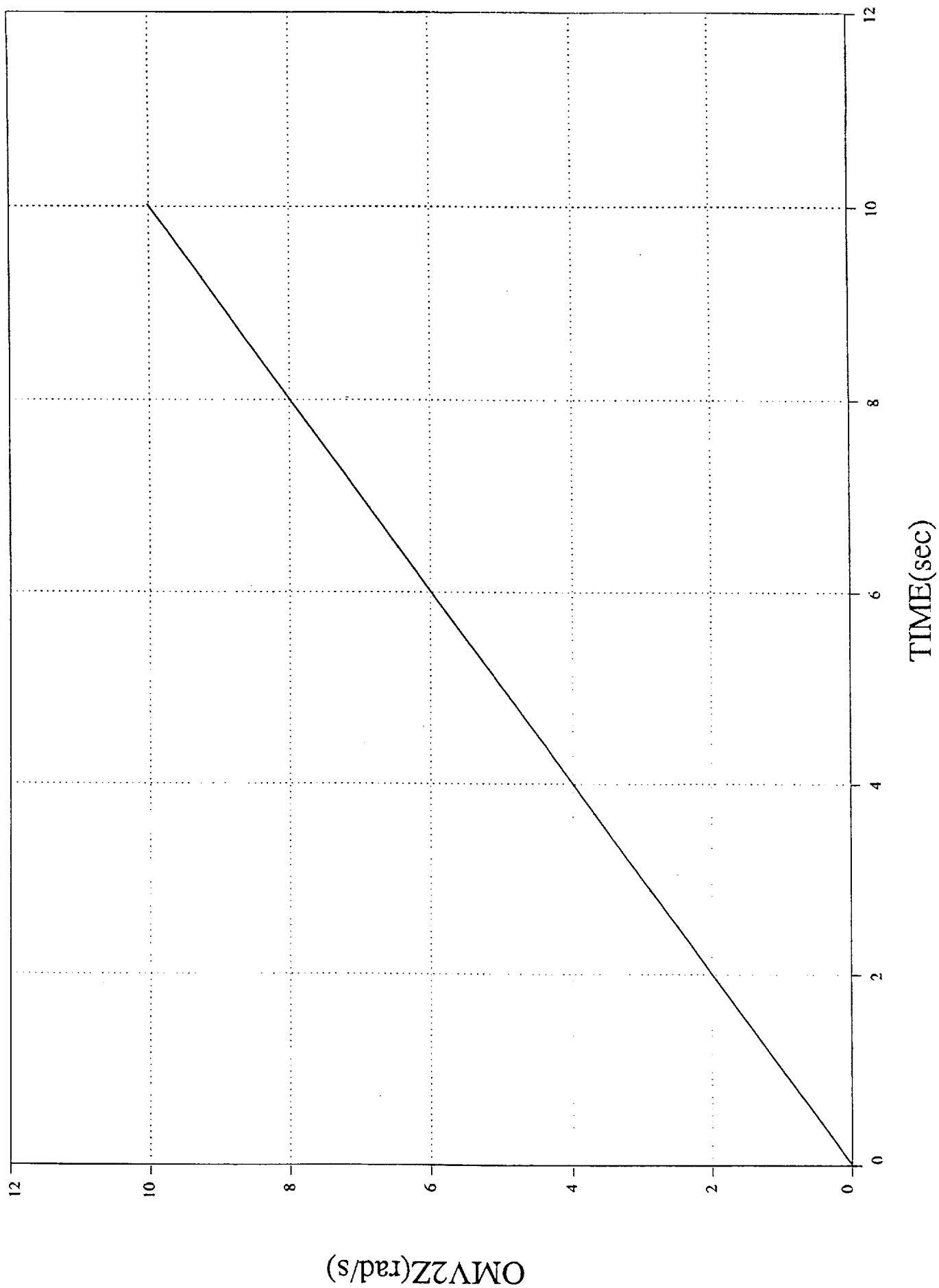




TEST 6

# DI/2BODY Angular Velocity vs. Time

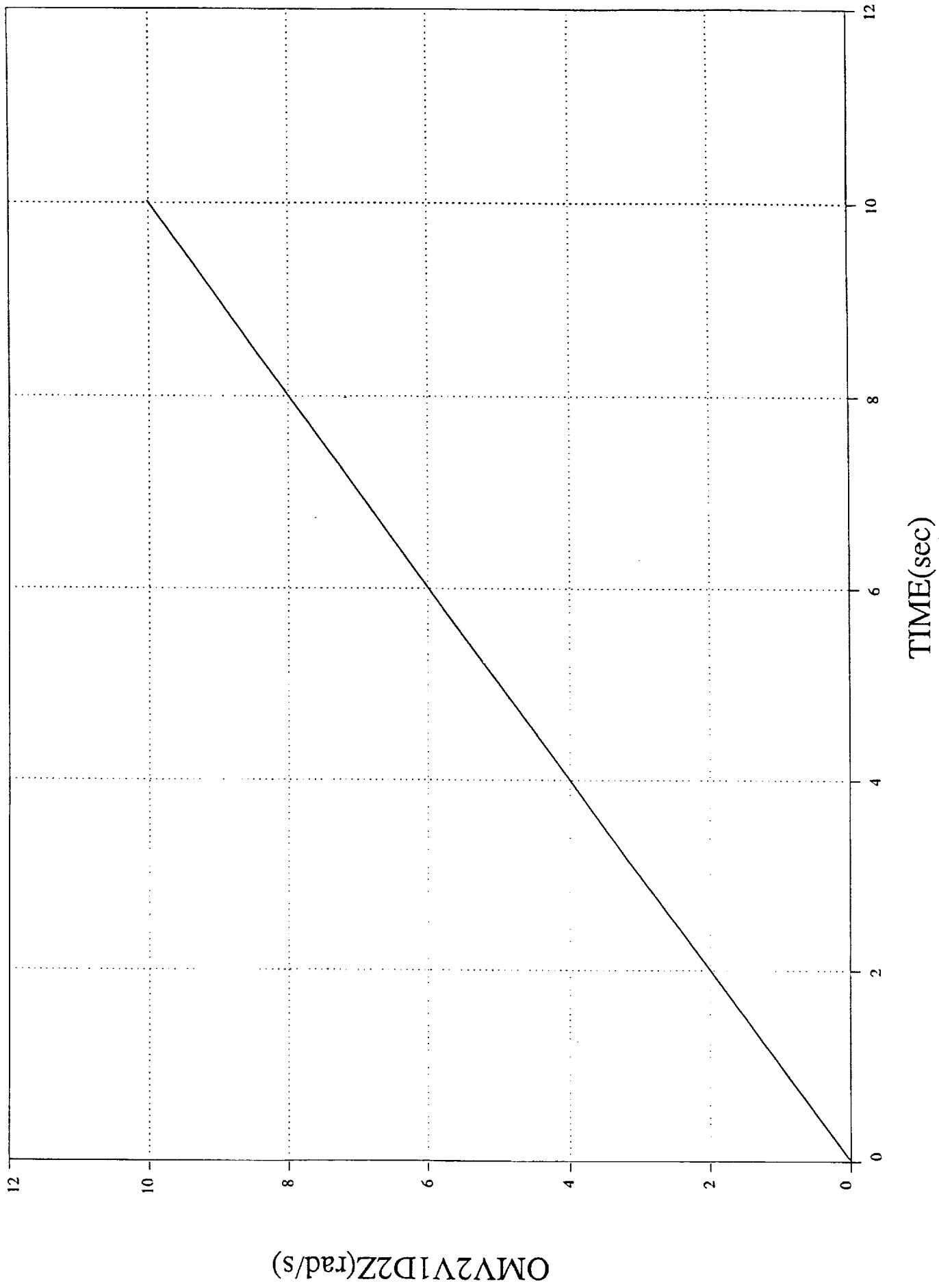
2bdt3  
1 0 — 0 33



TEST 6

# DI/2BODY Rel Ang Velocity vs. Time

2bddt3  
1 0 10

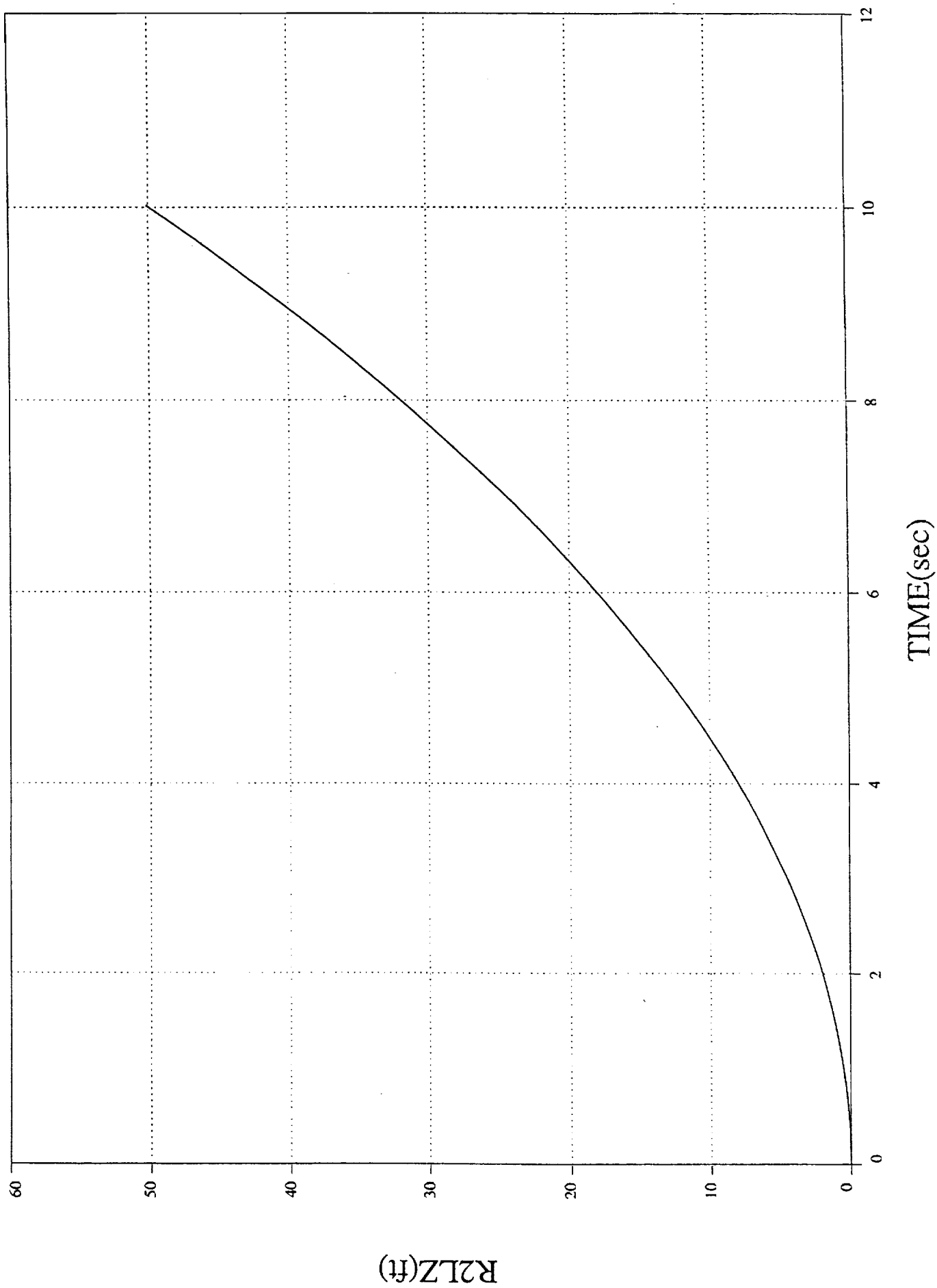


TEST 7

# DI/2BODY Position Body 2 vs. Time

2bdcf3

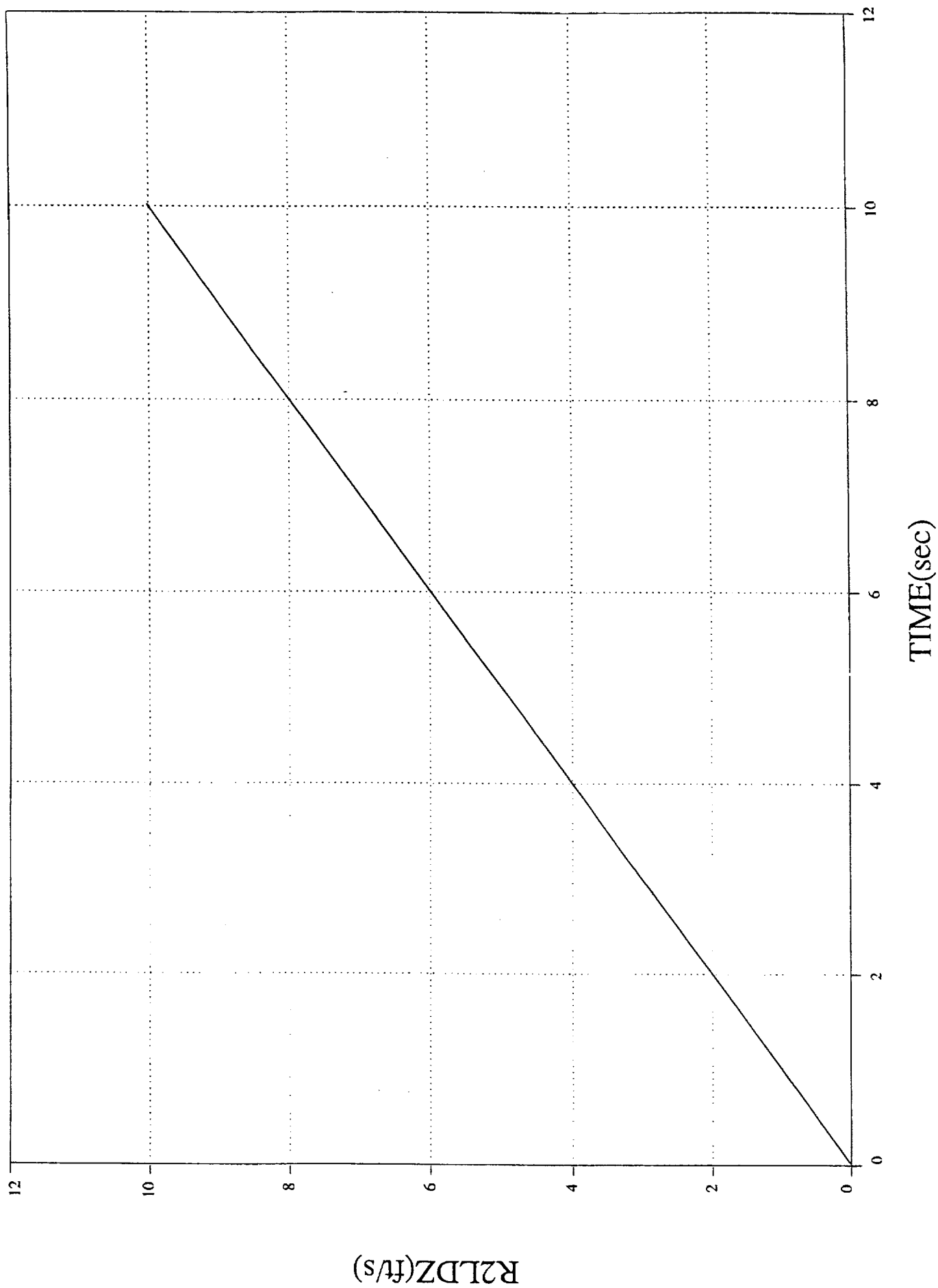
1 0 27



TEST 7

# DI/2BODY Velocity Body 2 vs. Time

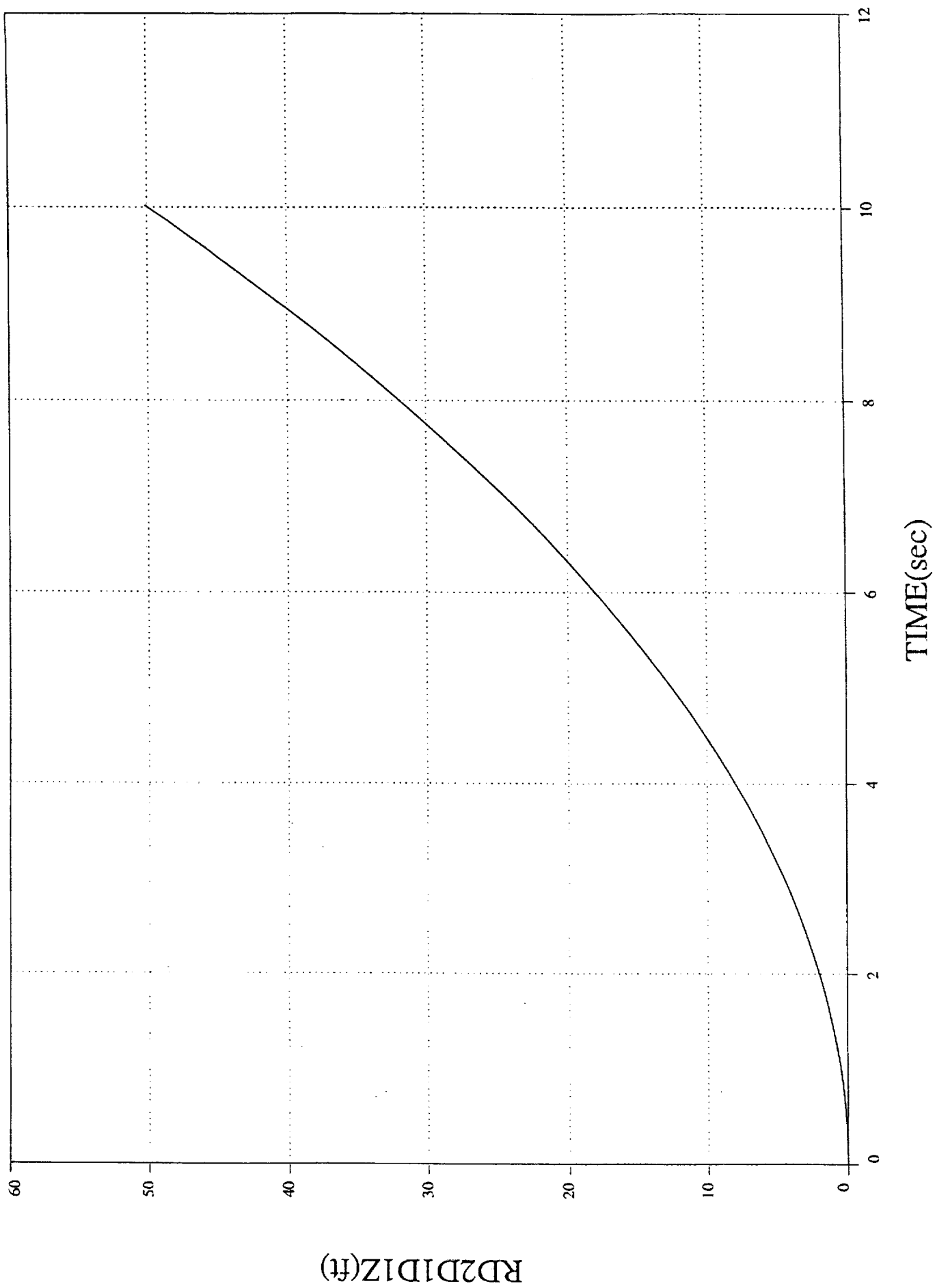
2bdcf3  
1 0 24



TEST 7

DI/2BODY Rel Pos Body1-2 vs. Time

2bdcf3  
1 0 — 0 4

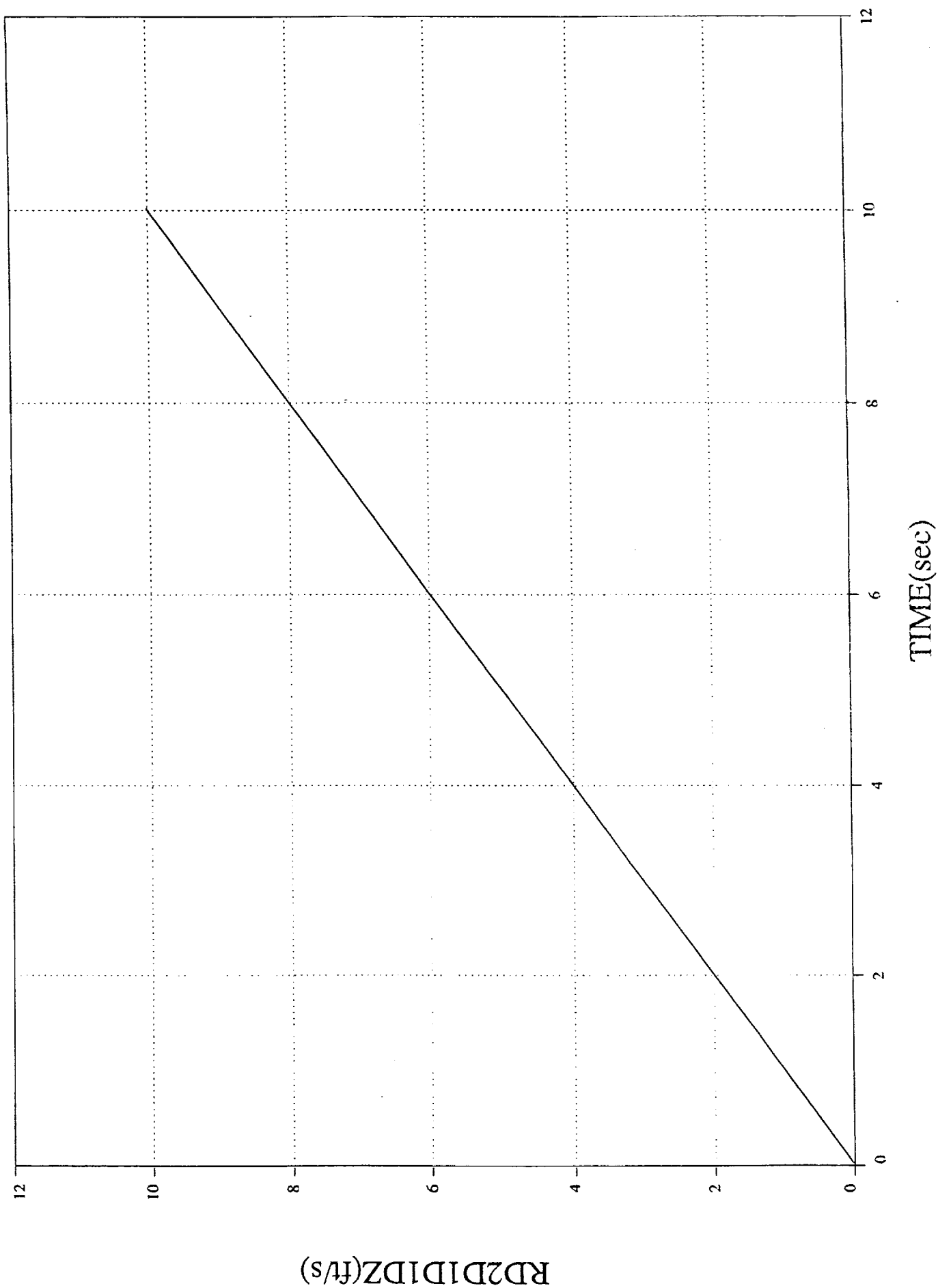


TEST 7

DI/2BODY Rel Vel Body1-2 vs. Time

2bdcf3

1 0 — 30

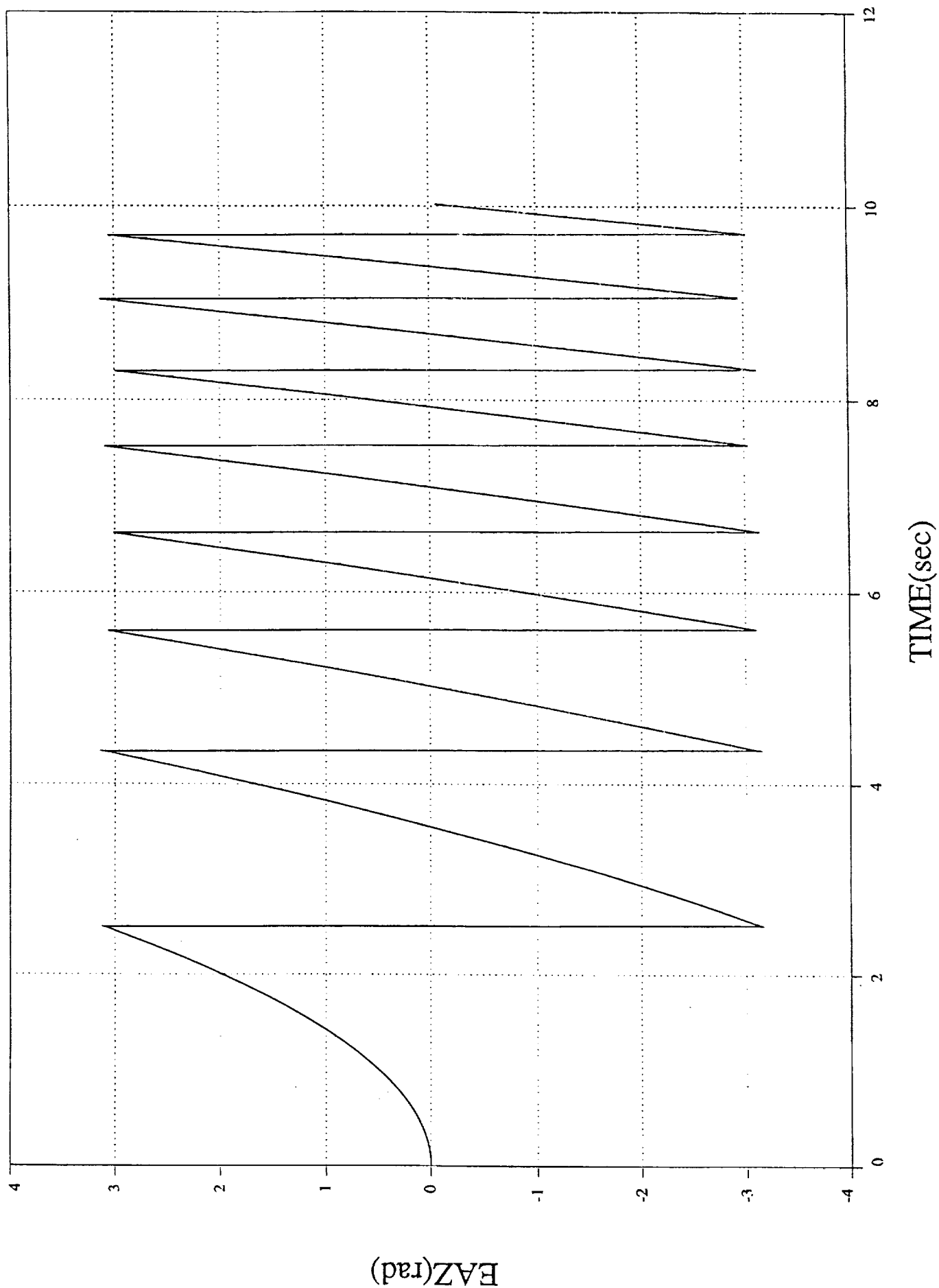


TEST 8

# DI/2BODY Euler Angle 2 vs. Time

2bdc13

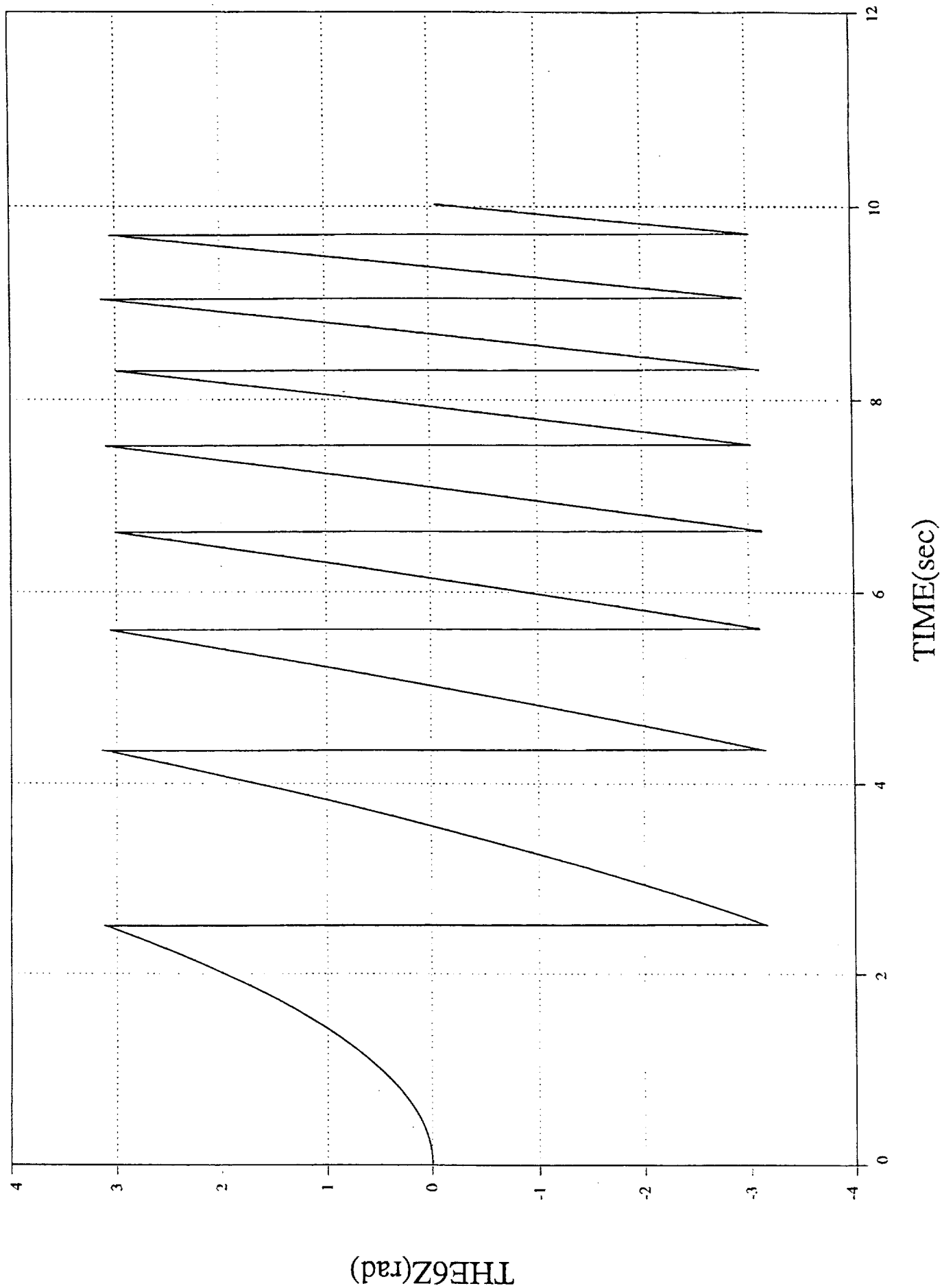
1 0 — 5



TEST 8

# DI/2BODY Rel Rotation vs. Time

2bdct3  
1 0 39



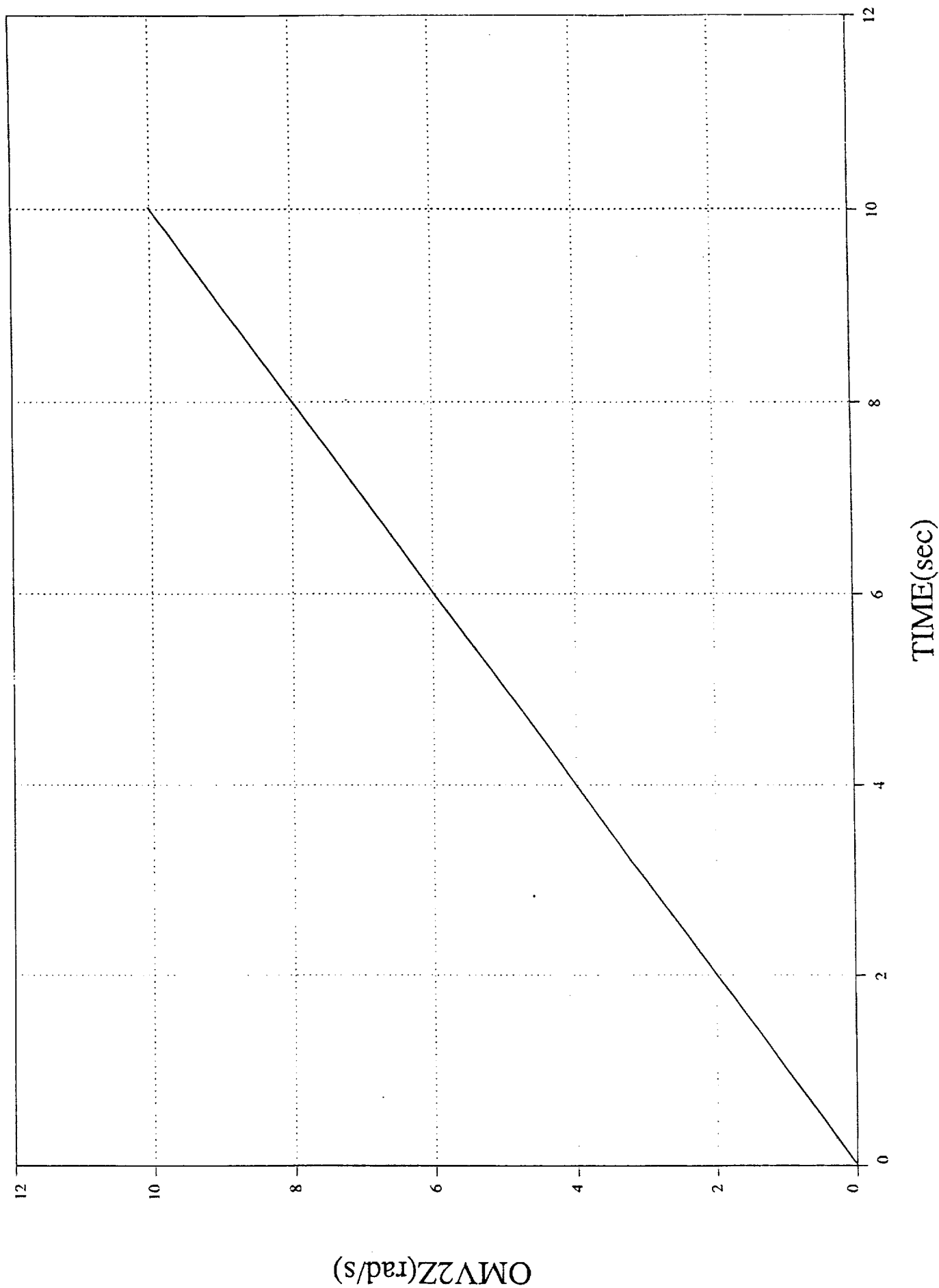


TEST 8

# DI/2BODY Angular Velocity vs. Time

2bdct3

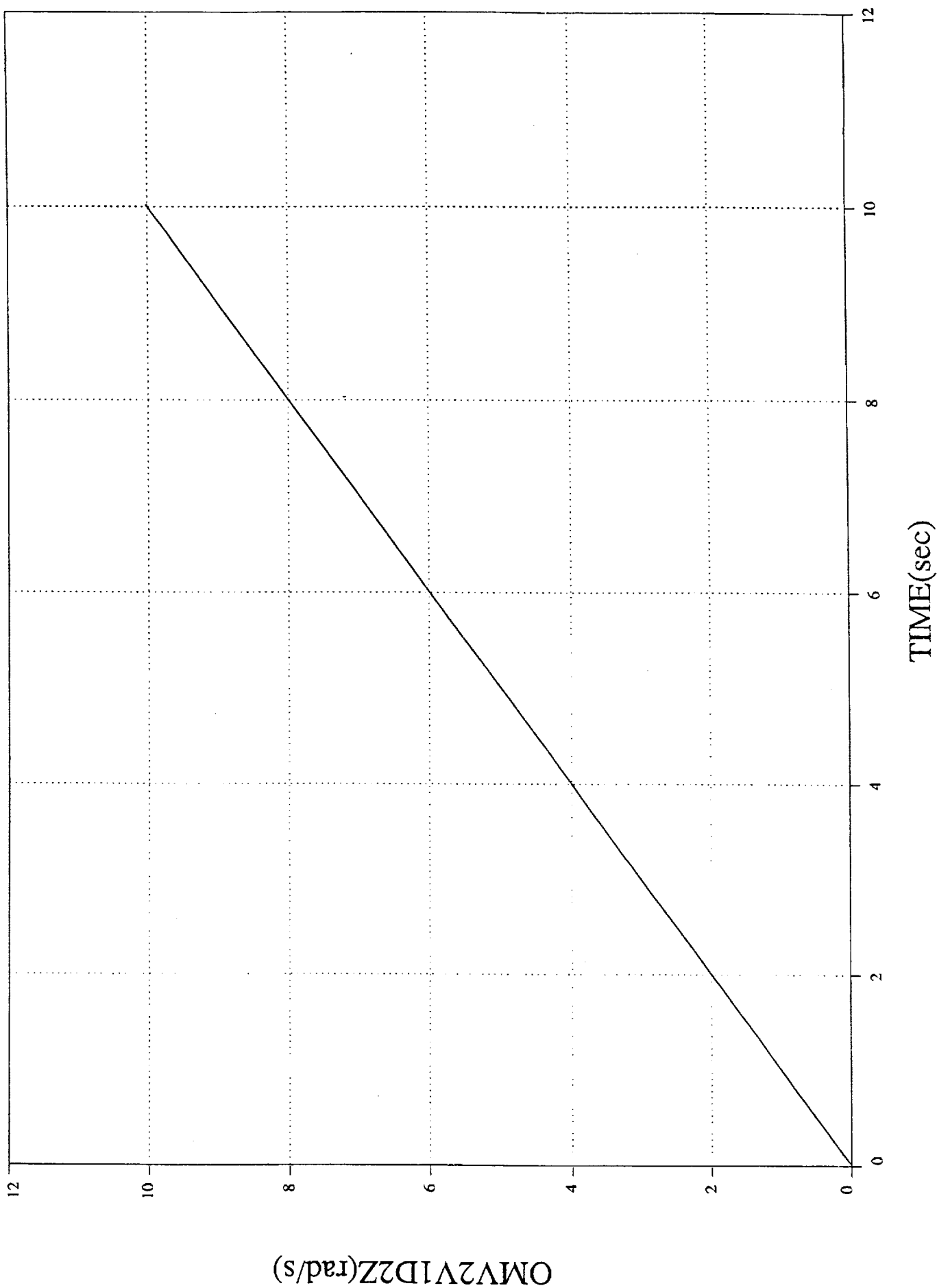
1 0 — 0 33



TEST 8

# DI/2BODY Rel Ang Velocity vs. Time

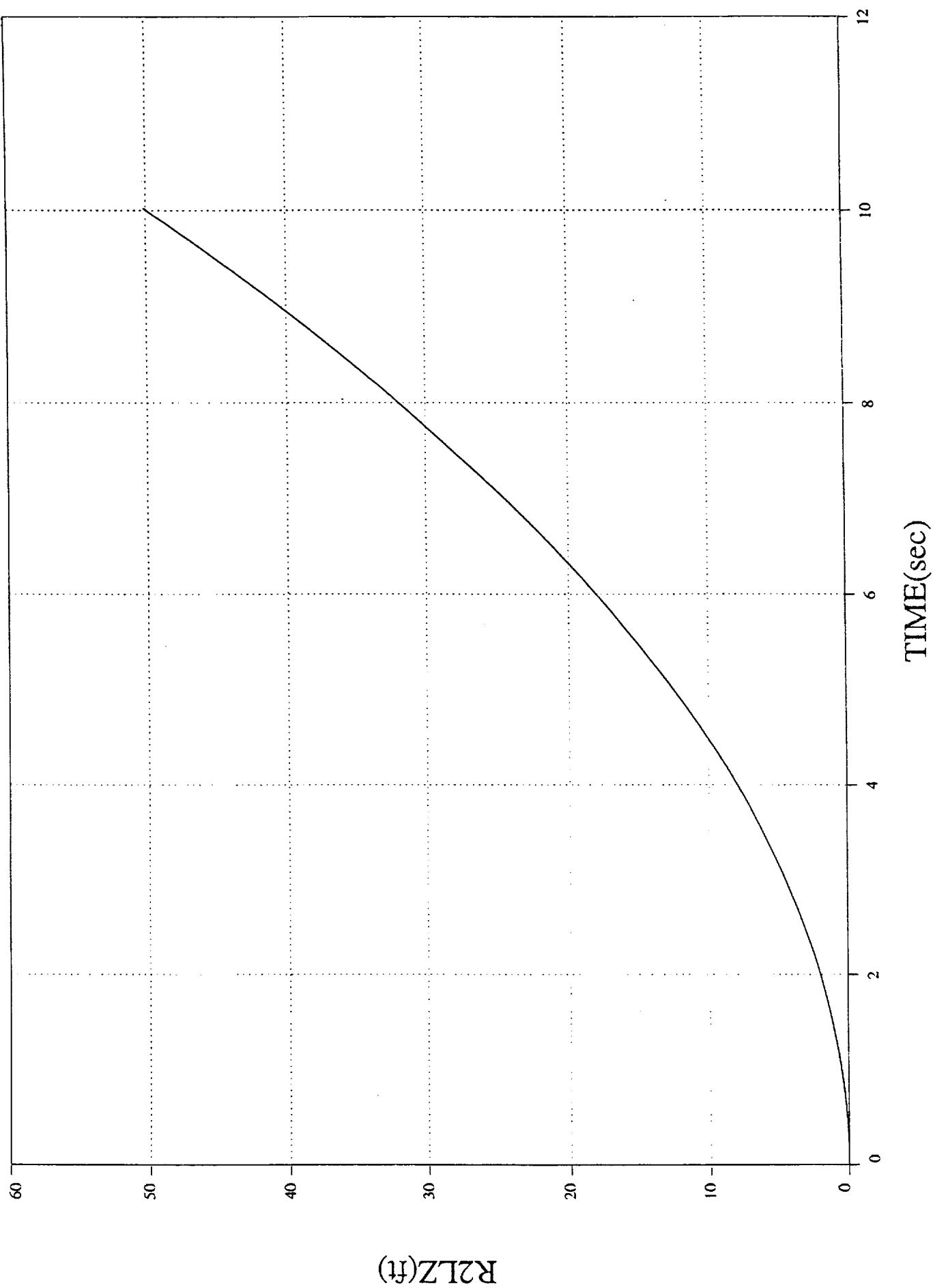
2bdct3  
1 0 — 10



TEST 7

# DI/2BODY Position Body 2 vs. Time

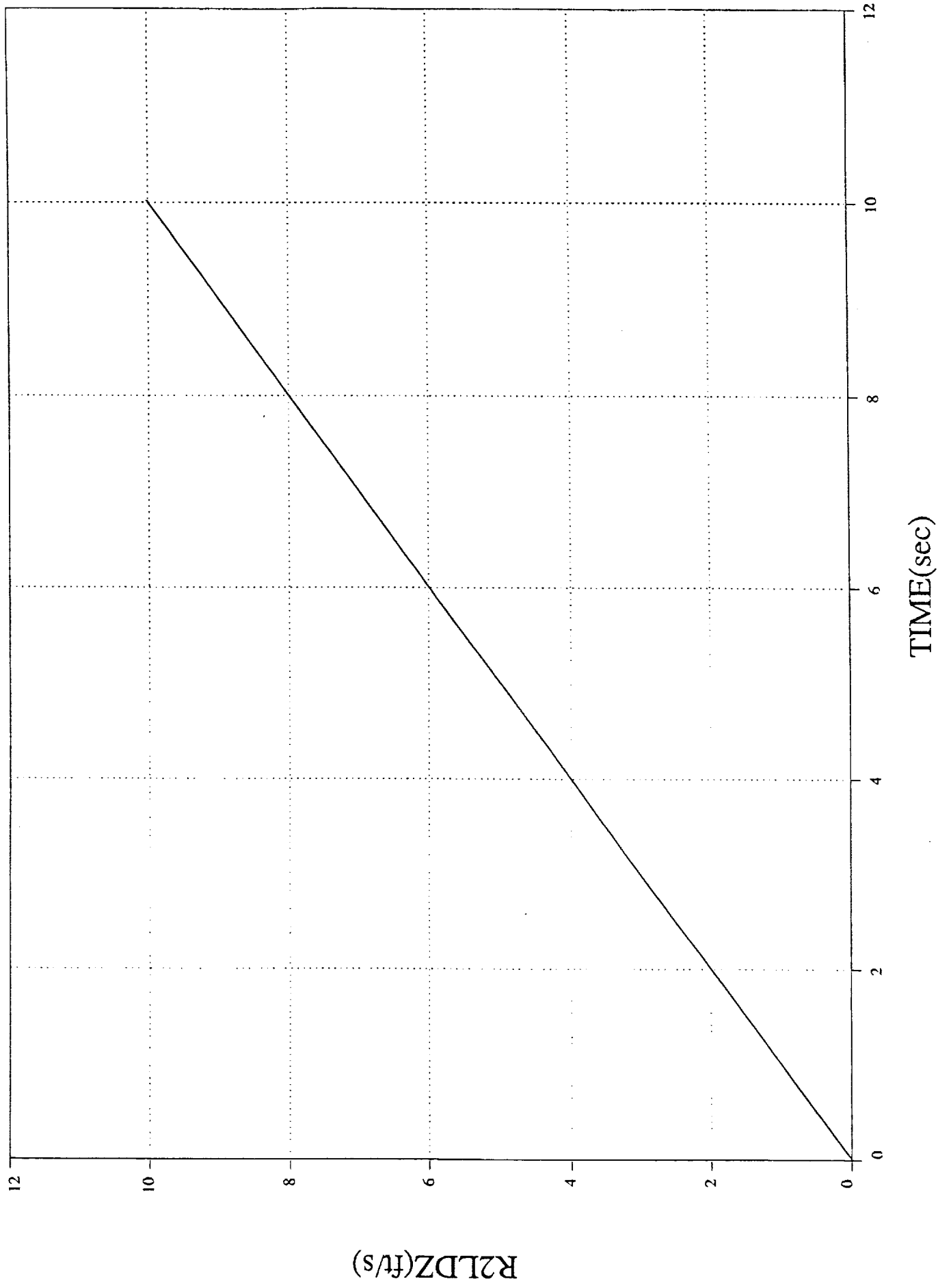
2bdcf3v2  
1 0 — 0 27



TEST 9

# DI/2BODY Velocity Body 2 vs. Time

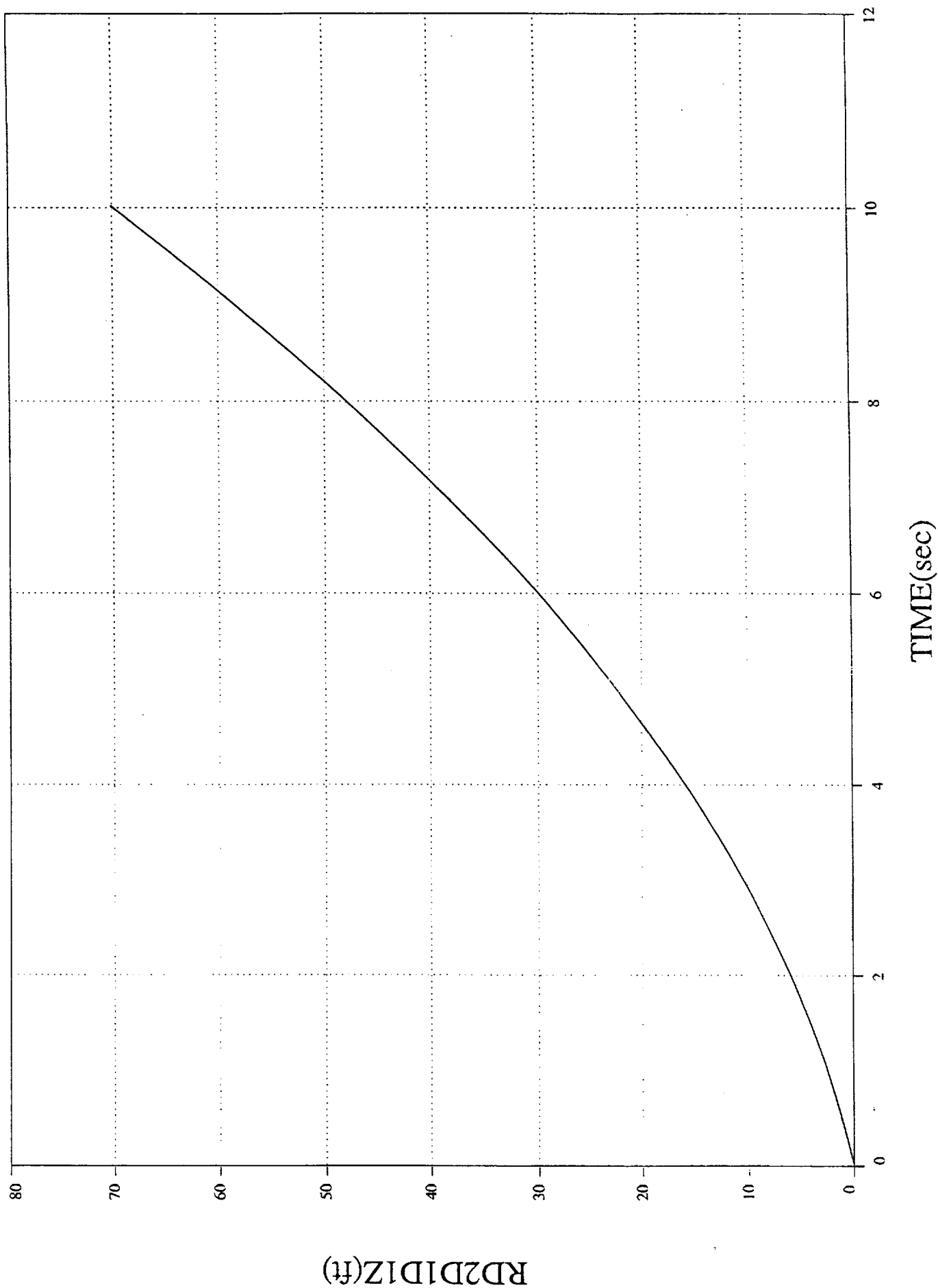
2bdef3v2  
1 0 — 24



TEST 9

# DI/2BODY Rel Pos Body1-2 vs. Time

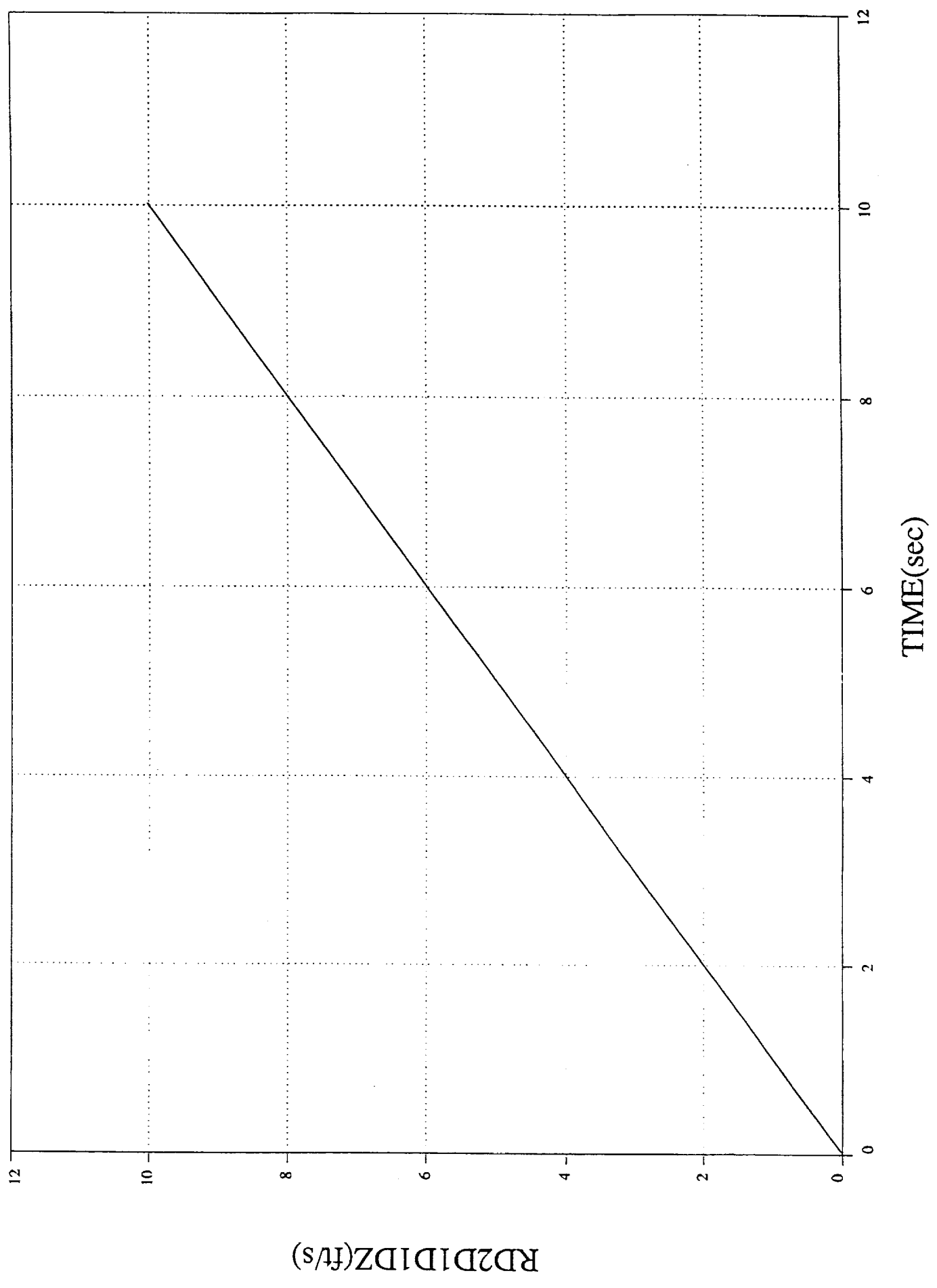
2bdcf3v2  
1 0 — 4



TEST 9

# DI/2BODY Rel Vel Body1-2 vs. Time

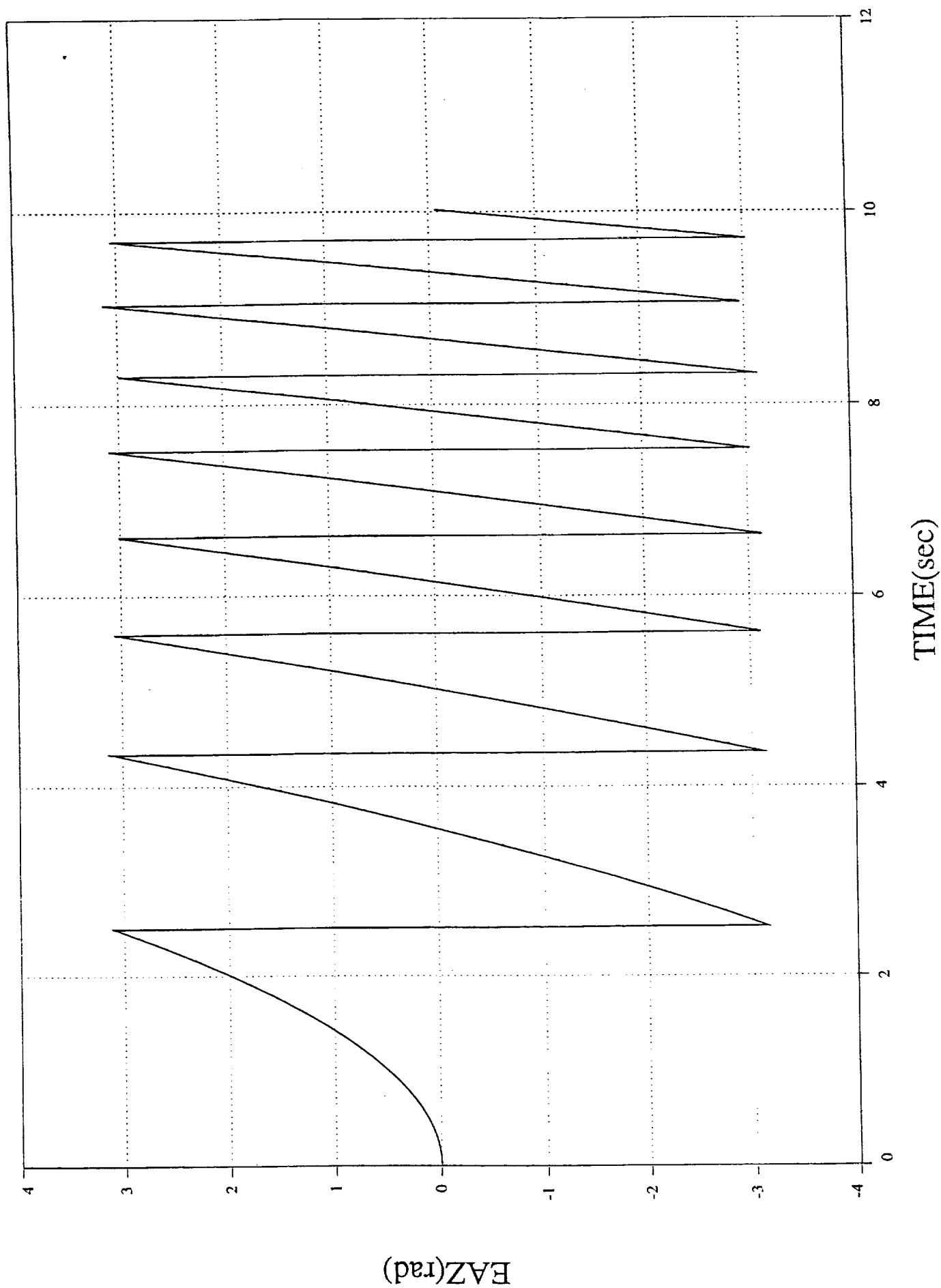
2bdcf3v2  
1 0 30



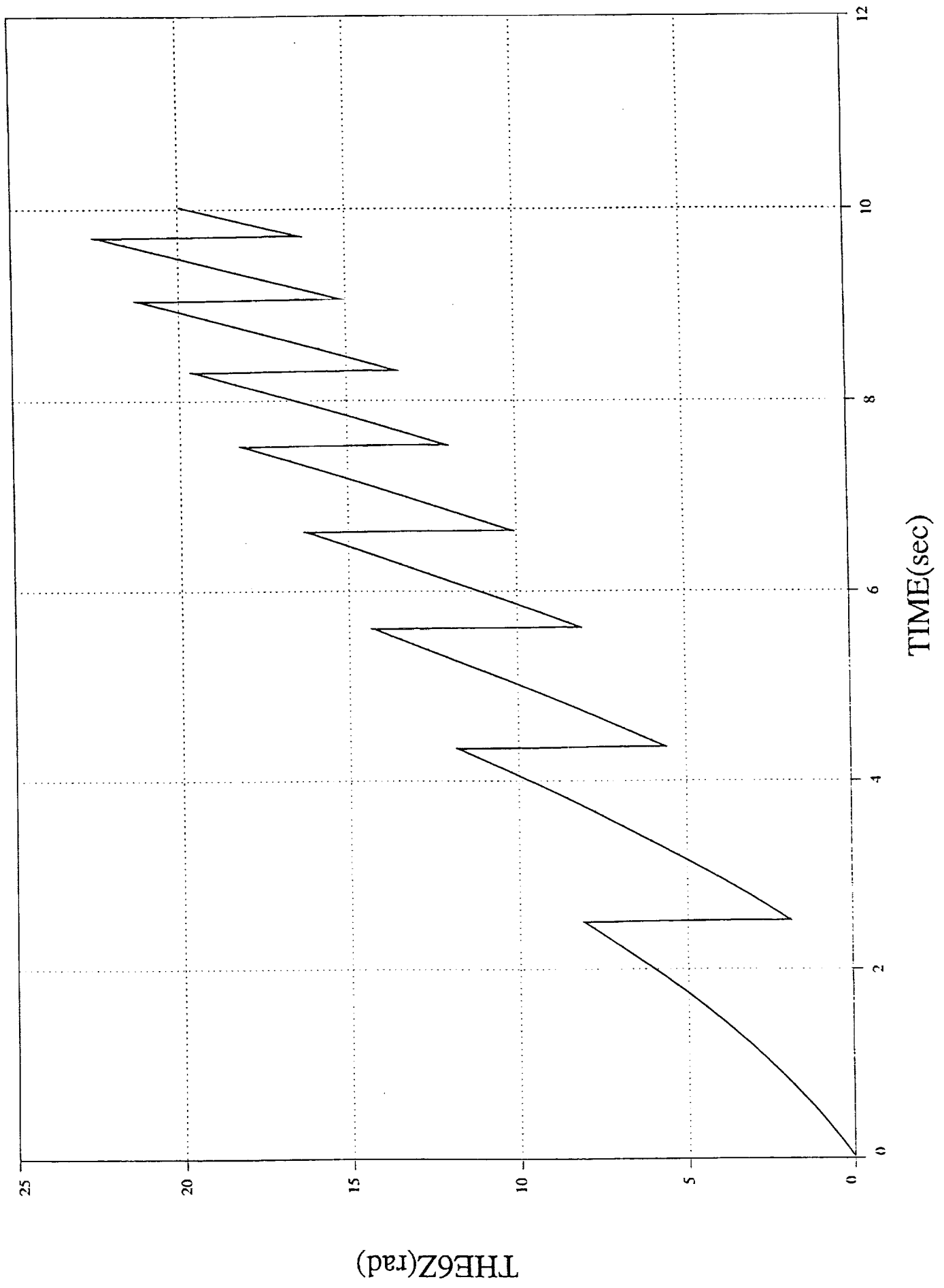
TEST 10

# DI/2BODY Euler Angle 2 vs. Time

2bdct3v2  
1 0 5



## DI/2BODY Rel Rotation vs. Time

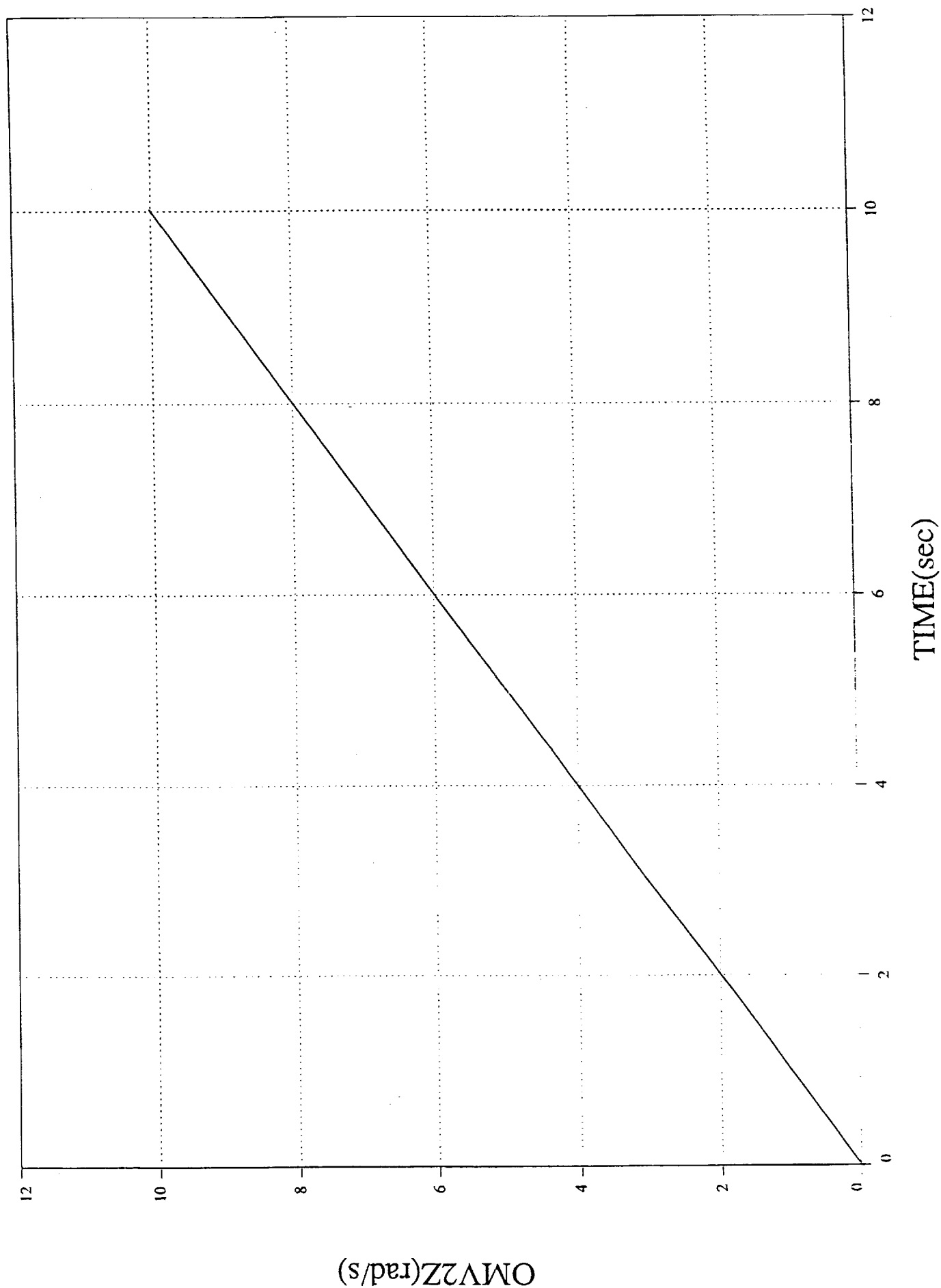
2bdct3v2  
1 0 — 39



TEST 10

# DI/2BODY Angular Velocity vs. Time

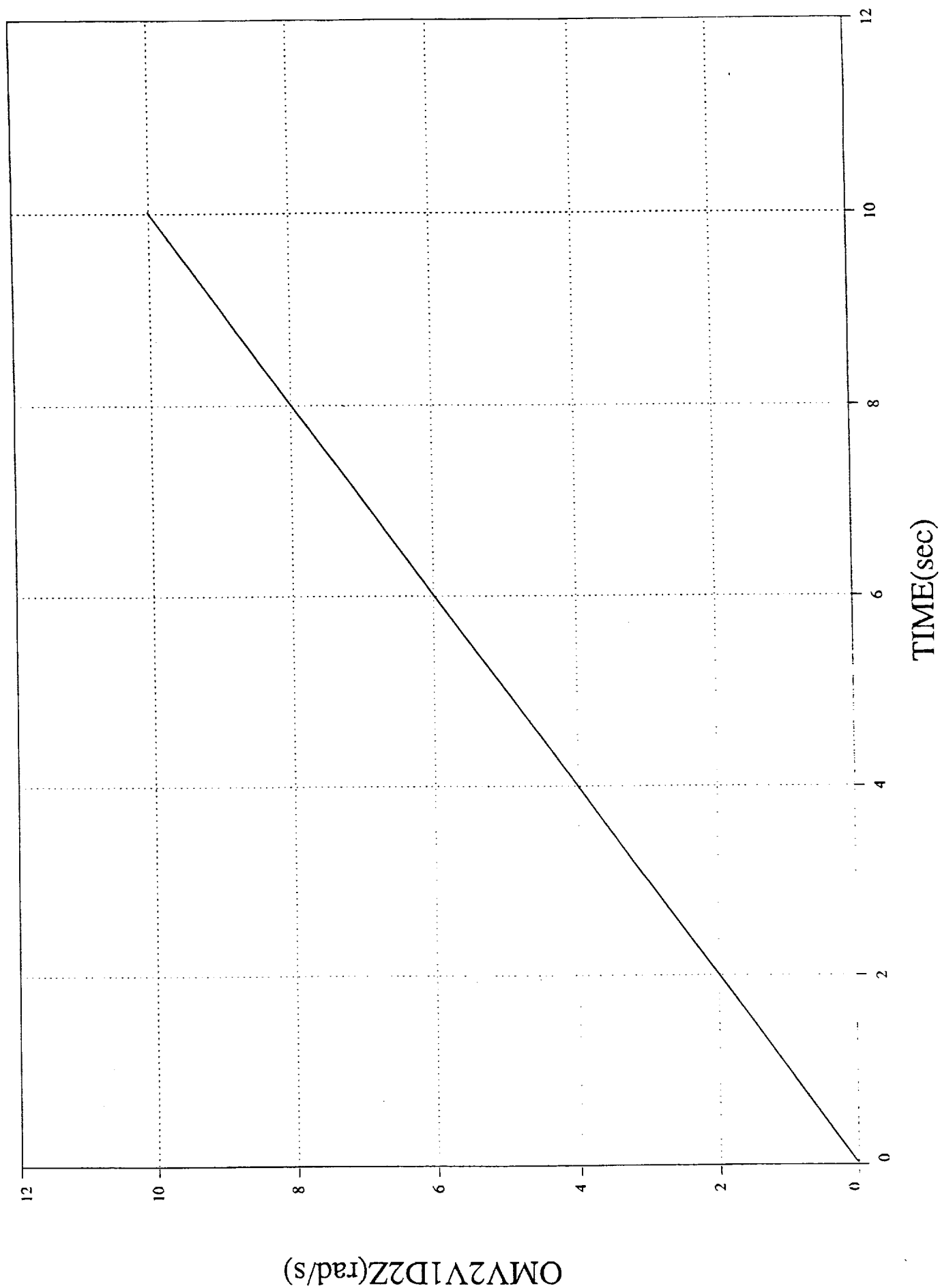
2bdc3v2  
1 0 — 0 33



TEST 10

# DI/2BODY Rel Ang Velocity vs. Time

2bdct3v2  
1 0 — 10

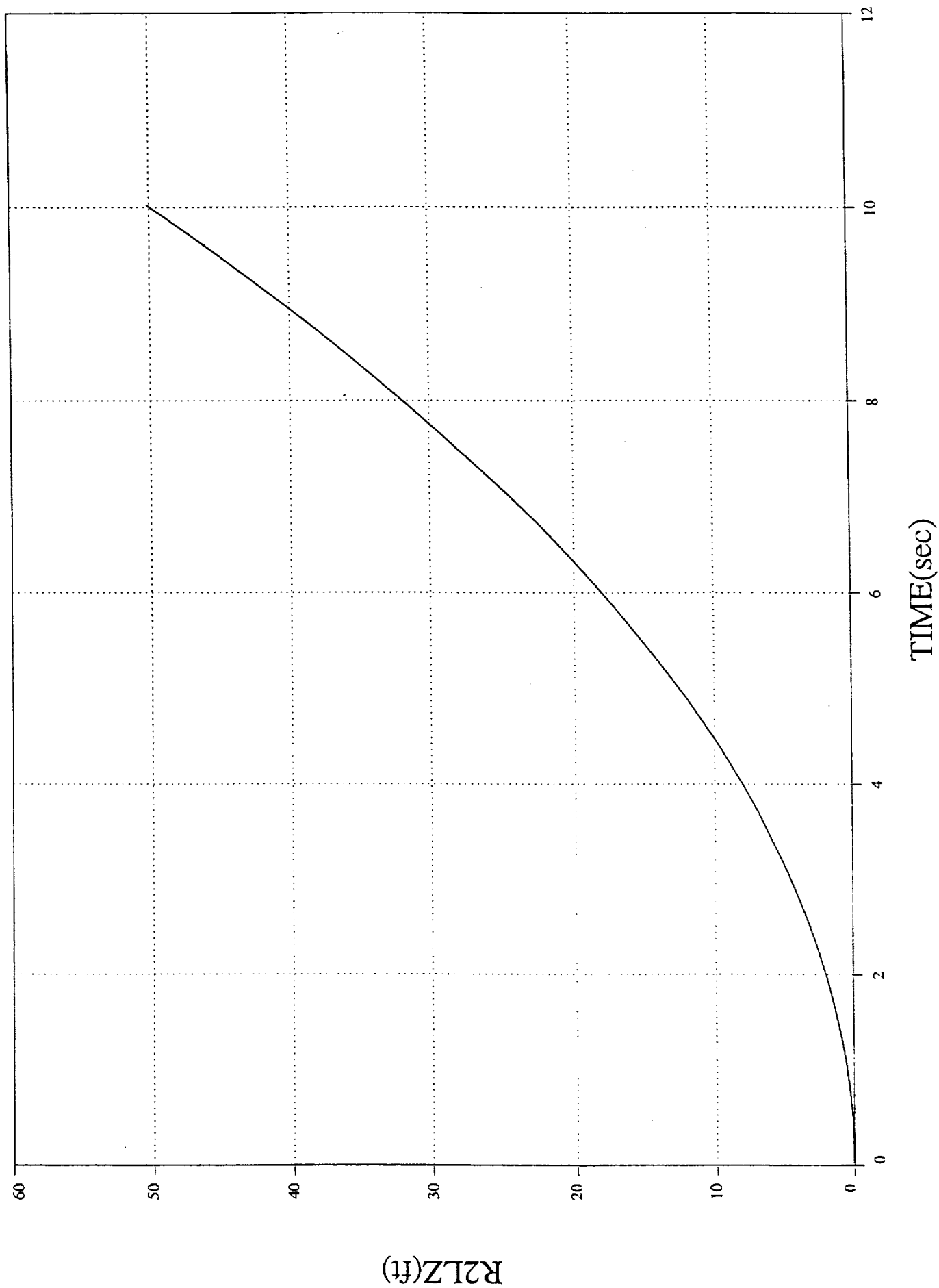


TEST II

# DI/2BODY Position Body 2 vs. Time

2bdlcf3v4

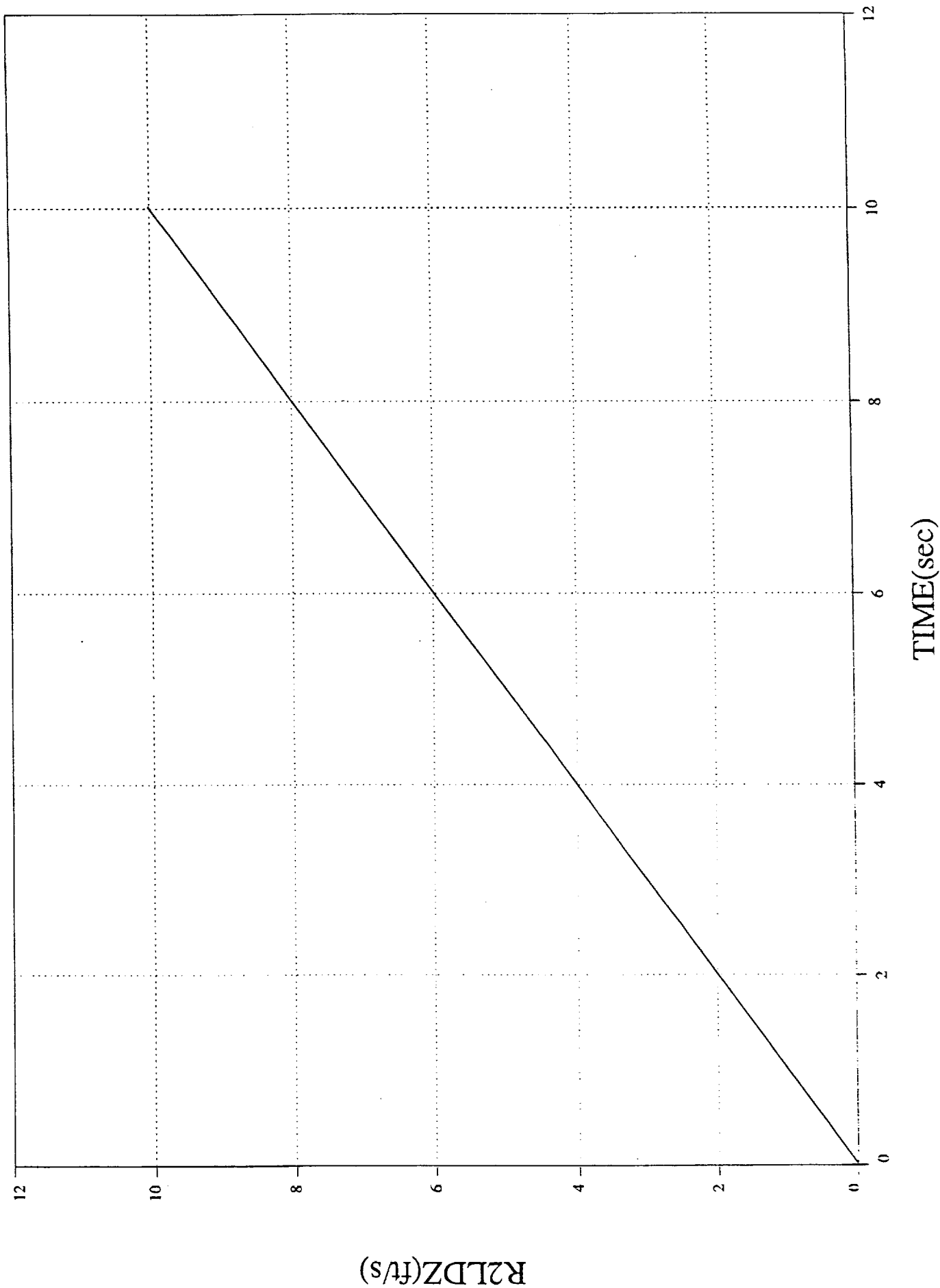
1 0 — 27



TEST K

# DI/2BODY Velocity Body 2 vs. Time

2bdcf3v4  
1 0 — 0 24

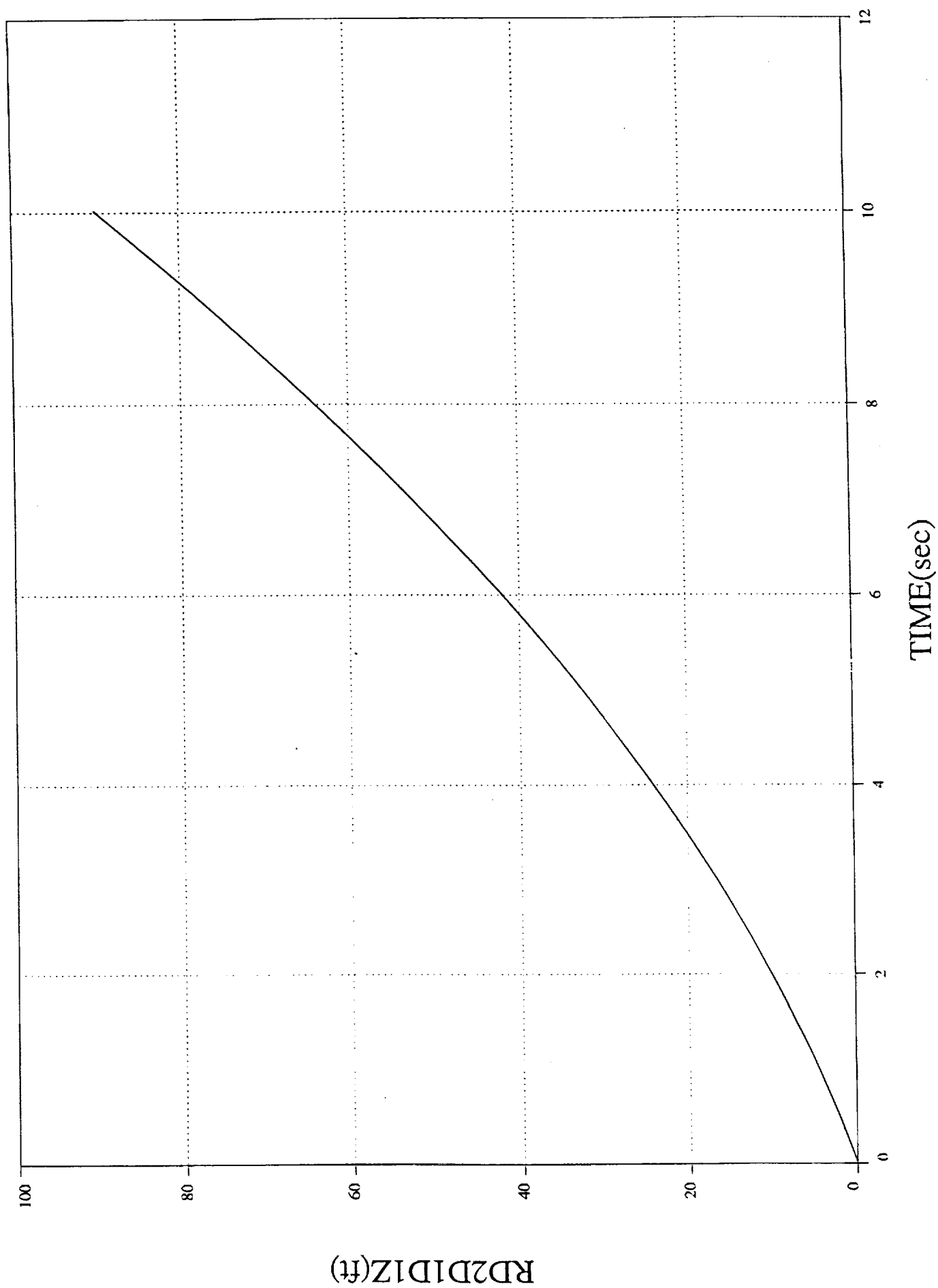


Test II

DI/2BODY Rel Pos Body1-2 vs. Time

2bdcf3v4

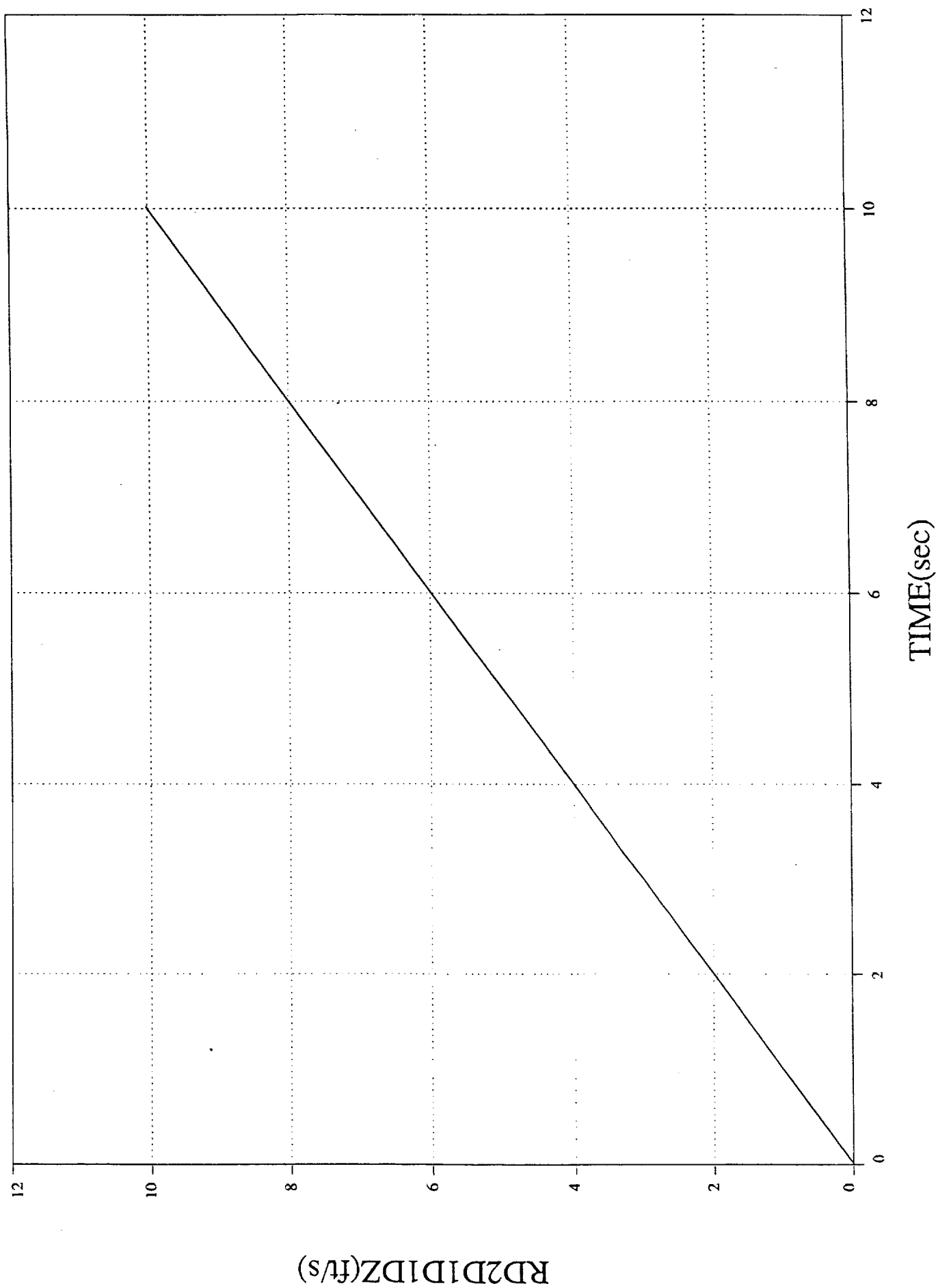
1 0 4



TEST II

# DI/2BODY Rel Vel Body1-2 vs. Time

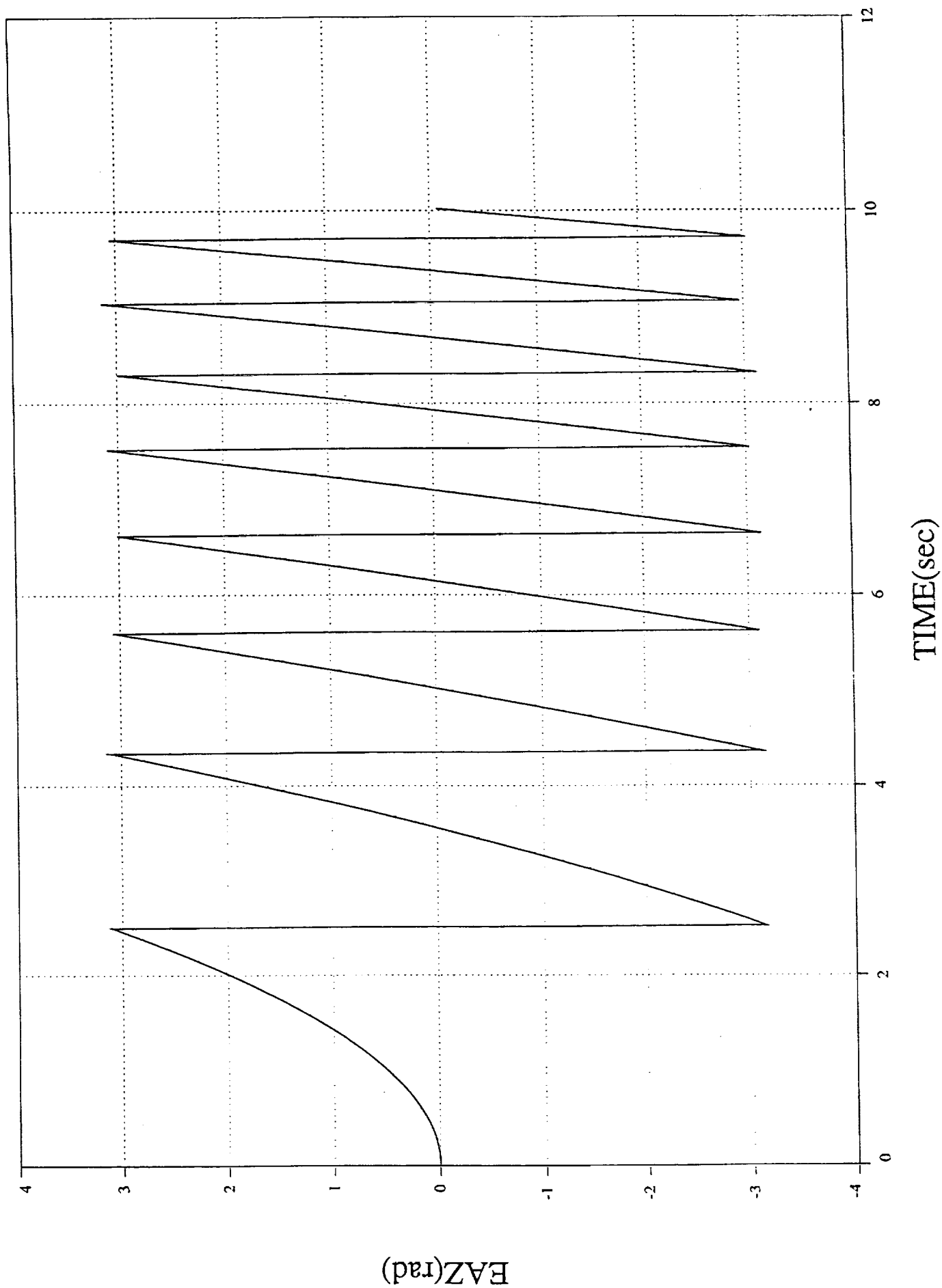
2bdcf3v4  
1 0 — 30



TEST 12

# DI/2BODY Euler Angle 2 vs. Time

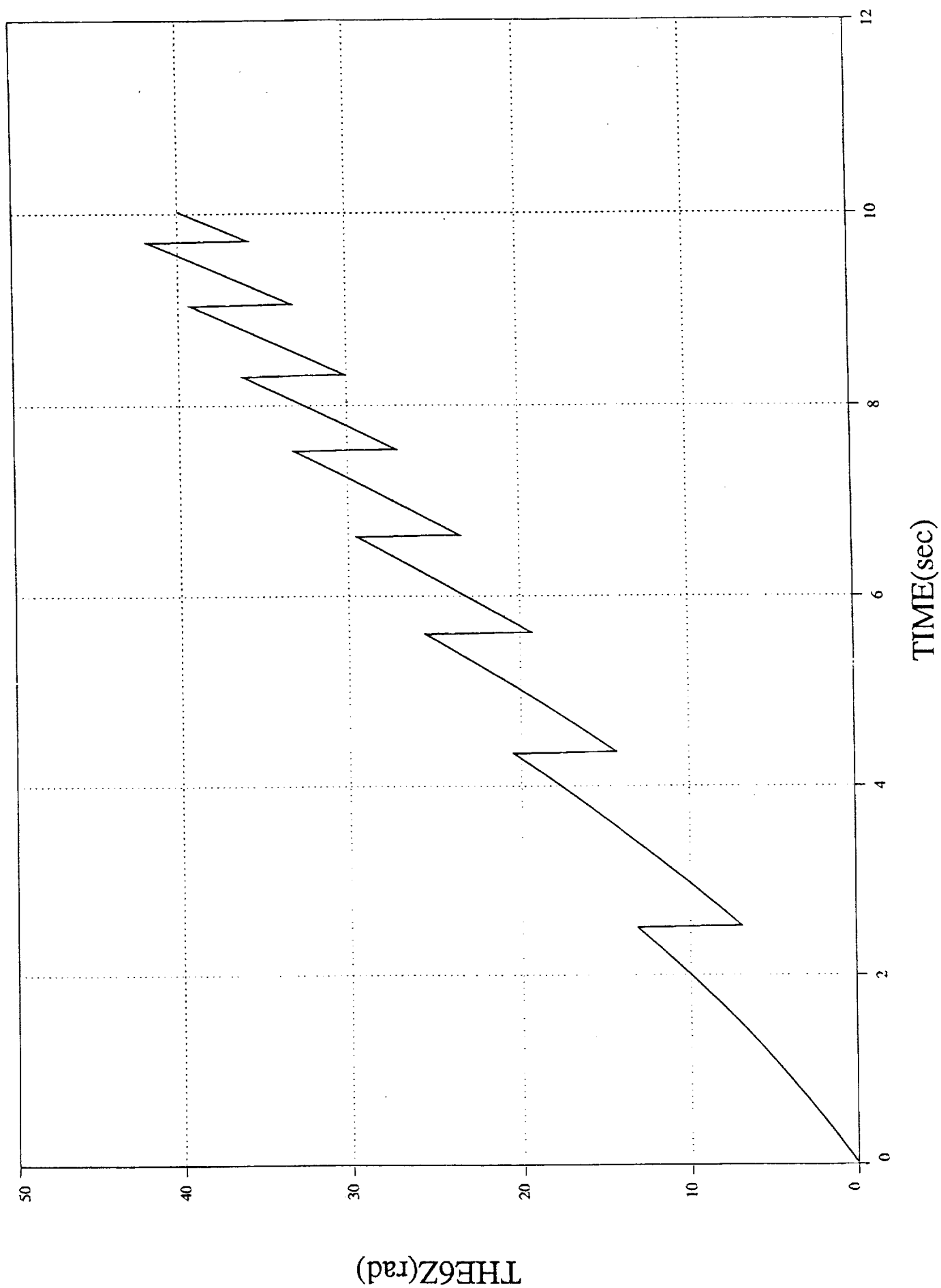
2bdct3v4  
1 0 — 5



TEST 12

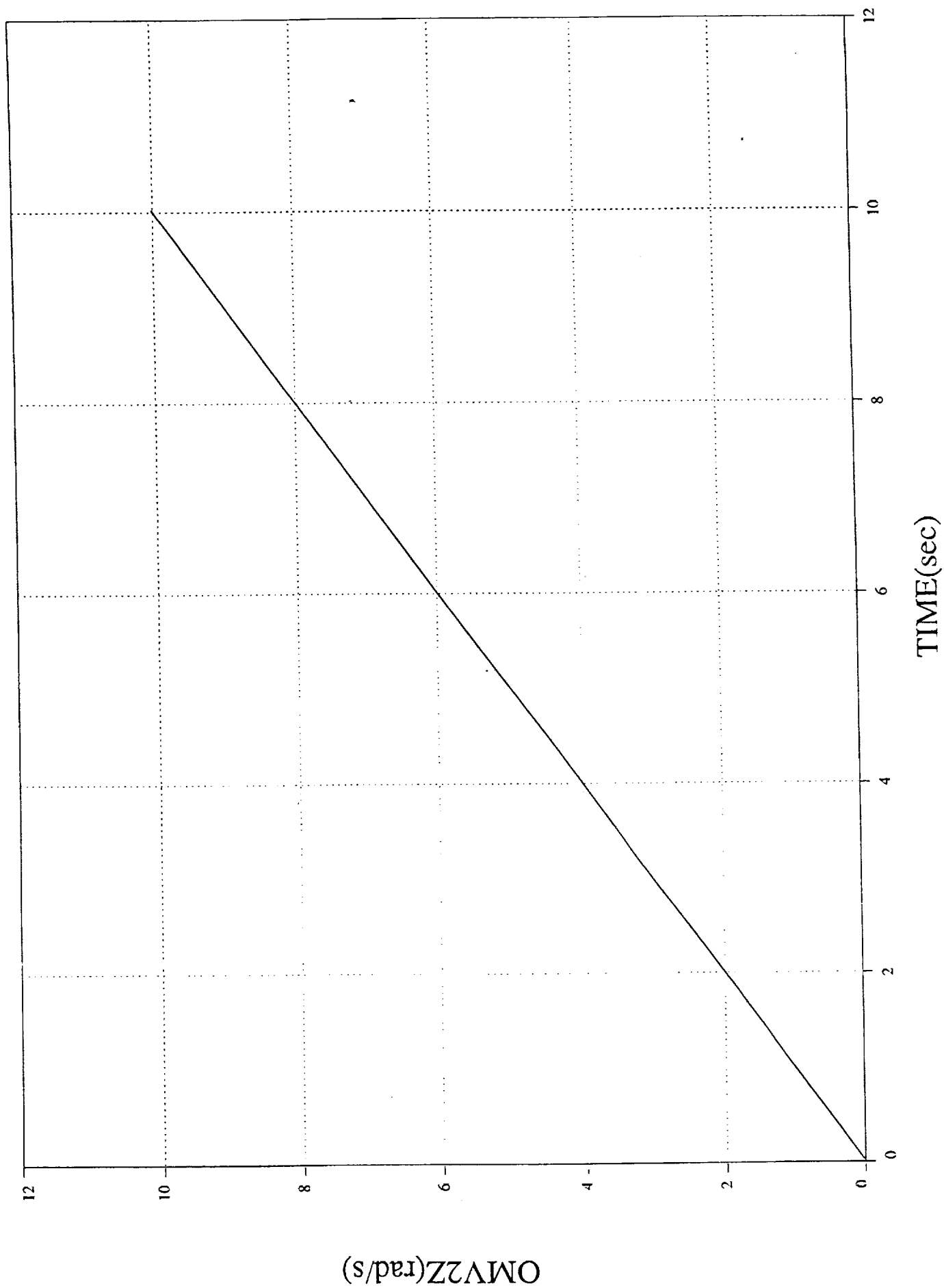
# DI/2BODY Rel Rotation vs. Time

2bdc3v4  
1 0 — 39





## DI/2BODY Angular Velocity vs. Time

2bdct3v4  
1 0 — 33

## Reference 9

DCI TM#011398-2  
Deberth Code Modifications and Test Runs  
January 13, 1998

## Dynamic Concepts Technical Memorandum #011398-2

To: Mr. Ronnie Francis, bd Systems

From: Dynamic Concepts - Dr. Patrick Tobbe, bd Systems - Mr. Mike Ekbundit

Subject: DEBERTH Code Modifications and Test Runs

Date: January 13, 1998

### 1.0 Introduction

The hardware in the loop simulation utility DEBERTH is hosted on the SGI Challenge machine UQBAR. DEBERTH simulates two connected bodies separating after docking. The simulation code was modified to function with new shared memory regions and was tested to ensure that the changes were made properly. A temporary driver was created to test the DEBERTH code independent of the new real time executive routine. The test cases include uniaxial forces and torques along each of the cardinal axes without damping and a uniaxial force case with damping. This memorandum will describe the test cases and software modifications to DEBERTH.

### 1.1 Modifications

DEBERTH was ported to UQBAR from the Alliant machine and is an option of the dynamics process in the latest version of ROCKET and 2BODY. All of the include files in DEBERTH were combined to form a new structure DB. This new structure was added to the argument lists of all appropriate subroutine statements and calls to the subroutines. Finally, seven test cases were run with the modified code.

### 2.0 Software Modifications

All of the include files based in DEBERTH with the prefix "cmmn" were combined into a new structure, "s\_db.inc", shown in Table 1. The prefix "DB." was added to all of the appropriate variables in DEBERTH and the structure was added to the argument lists of the appropriate SUBROUTINE, CALL, and ENTRY statements.

|                |
|----------------|
| s_db.inc       |
| cmmndb.inc     |
| cmmn1db.inc    |
| cmmn2db.inc    |
| cmmnbdb.inc    |
| cmmncompdb.inc |

TABLE 1 - Include files combined into structure DB

### 3.0 Example Runs

Seven cases were run with the DEBERTH code including uniaxial forces and torques with no damping about each of the cardinal axes and uniaxial force and torque with damping. The loads were applied one at a time in separate tests. Table 2 illustrates the run descriptions for each of the test cases.

| TEST CASE | LOAD TYPE | LOAD AXIS | DAMPING VALUE | LOAD      |
|-----------|-----------|-----------|---------------|-----------|
| 1         | FORCE     | X         | 0             | 100lb     |
| 2         | FORCE     | Y         | 0             | 100lb     |
| 3         | FORCE     | Z         | 0             | 100lb     |
| 4         | TORQUE    | X         | 0             | 100 ft-lb |
| 5         | TORQUE    | Y         | 0             | 5 ft-lb   |
| 6         | TORQUE    | Z         | 0             | 100 ft-lb |
| 7         | FORCE     | X         | 50            | 100lb     |

TABLE 2 - DEBERTH test cases

### 3.1 Prediction of Test Results

The single axis tests were designed to have easily verifiable results by using mass, inertia, and load values which greatly simplify the equations of motion. If body one is assumed to be stationary then the motion of body two is also the relative motion between the bodies.

In cases where there is no damping, single axis translational and angular acceleration terms  $a_2$  and  $\alpha_2$  are given by

$$F_2 = m_2 a_2$$

$$T_2 = I_2 \alpha_2$$

Therefore the position of body two can be found by

$$R_2 = v_o t + \frac{1}{2} a_2 t^2$$

for constant force cases. The corresponding velocity of body two with zero initial rate is

$$\dot{R}_2 = a_2 t$$

Similarly, because the applied torques are constant and uniaxial, angular position can be found by

$$\theta_2 = \omega_0 t + \frac{1}{2} \alpha_2 t^2$$

When  $\omega_0$  is zero, the angular velocity can be found by

$$\dot{\theta}_2 = \alpha_2 t$$

The test cases were set up such that  $F_2 = 10\text{lb}$ ,  $I_2 = 10\text{lb-ft-sec}^2$ ,  $T_2 = 10\text{ ft-lb}$  and  $m_2 = 10\text{lb}$ . This forces  $a_2$  and  $\alpha_2$  to be  $1\text{ ft/s}^2$  and  $1\text{ rad/s}^2$  respectively and leads to

$$\theta_2 = R_2 = \frac{1}{2} t^2$$

and

$$\dot{\theta}_2 = \dot{R}_2 = t$$

where  $R_2$  and  $\dot{R}_2$  have units of ft and ft/s respectively and  $\theta_2$  and  $\dot{\theta}_2$  have units of rad and rad/s respectively.

However, in the damping case, the equation of motion becomes a differential equation given by

$$m_2 \ddot{y}_2 = F_{XL} + F_{2CL} - T_{DAMP}(\dot{y}_2)$$

where  $m_2$  is the mass of body two,  $\ddot{y}_2$  is the acceleration of body two, and  $F_{XL}$  and  $F_{2CL}$  are forces acting on body two,  $T_{DAMP}$  is the translational damping term, and  $\dot{y}_2$  is the velocity of body two. The initial conditions are such that the initial position of body two is zero, the initial velocity of body two is  $10\text{ ft/s}$ ,  $m_2$  is  $100\text{lb}$ ,  $T_{DAMP}$  is  $50\text{ lb/s}$ , and  $F_{XL}$  and  $F_{2CL}$  are forced to zero. The differential equation can then be written as

$$100\ddot{y} + 50\dot{y} = 0$$

or

$$\ddot{y} + \frac{1}{2}\dot{y} = 0$$

The characteristic equation is then

$$r^2 + \frac{1}{2}r = 0$$

yielding the solutions  $r = 0, -1/2$ . Therefore the general solution to the differential equation is

$$y_g = C_1 + C_2 e^{-\frac{1}{2}t}$$

and

$$\dot{y}_g = -\frac{1}{2}C_2 e^{-\frac{1}{2}t}$$

Using the initial conditions  $y(0)=0$  and  $\dot{y}(0)=10$  to solve for  $C_1$  and  $C_2$  results in the following expressions:

$$y_g = 20 - 20e^{-\frac{1}{2}t}$$

and

$$\dot{y}_g = 10e^{-\frac{1}{2}t}$$

Table 3 illustrates the predicted values for the test cases at various time values

| TEST CASE | TDAMP | LOAD      | TIME   | POS   | VEL     | THETA    | OMV        |
|-----------|-------|-----------|--------|-------|---------|----------|------------|
| 1         | 0     | 100 lb    | 10 sec | 50 ft | 10 ft/s | N/A      | N/A        |
| 2         | 0     | 100 lb    | 10 sec | 50 ft | 10 ft/s | N/A      | N/A        |
| 3         | 0     | 100 lb    | 10 sec | 50 ft | 10 ft/s | N/A      | N/A        |
| 4         | 0     | 100 ft-lb | 10 sec | N/A   | N/A     | 50 rad   | 10 rad/s   |
| 5         | 0     | 5 ft-lb   | 4 sec  | N/A   | N/A     | -0.4 rad | -0.2 rad/s |
| 6         | 0     | 100 ft-lb | 10 sec | N/A   | N/A     | 50 rad   | 10 rad/s   |
| 7         | 50    | 100 lb    | 2 sec  | 12.64 | 3.68    | N/A      | N/A        |

TABLE 3 - Predicted results for test cases

Graphical test results appear in the appendix.

## 4.0 Conclusions

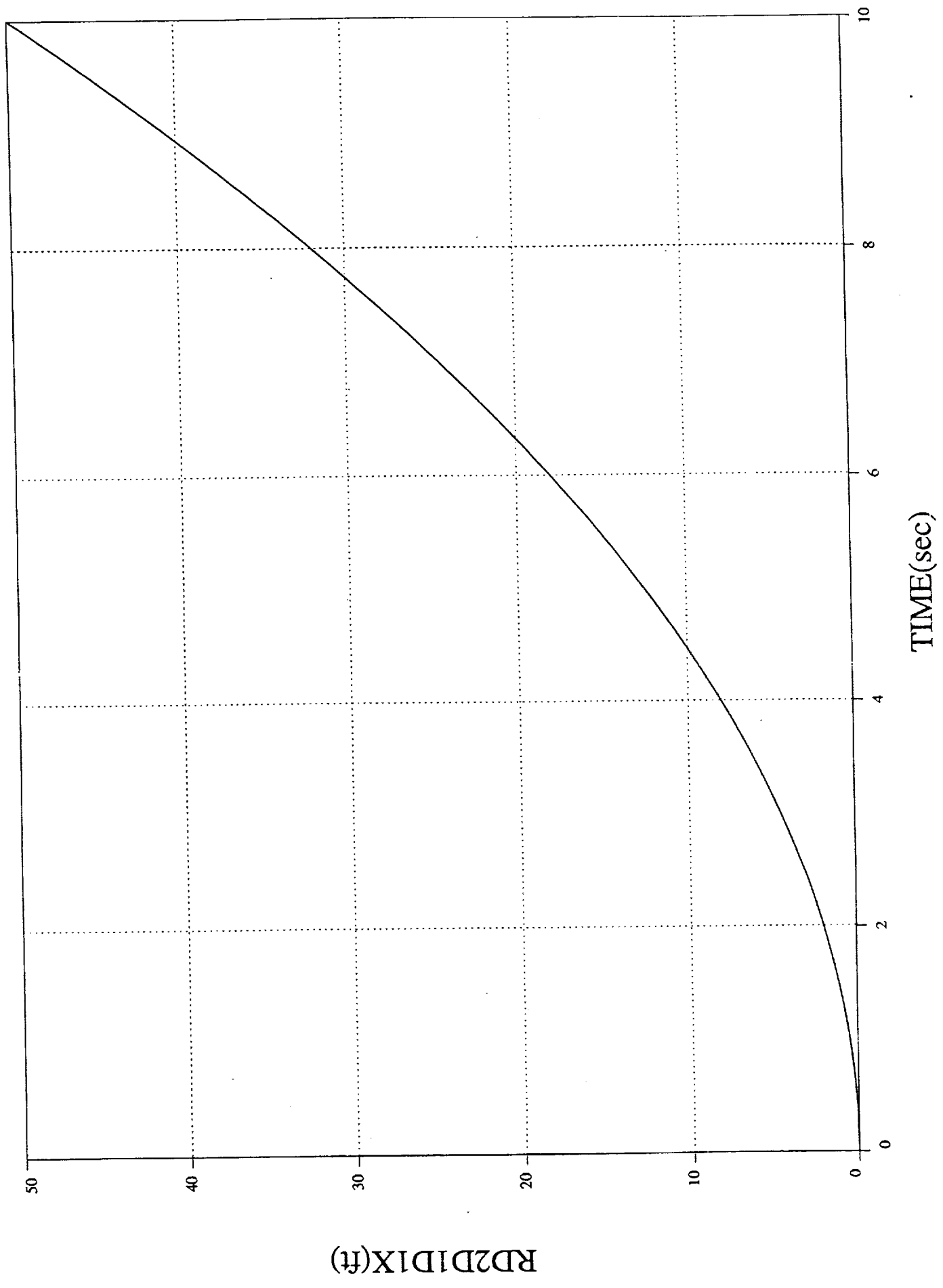
DEBERTH has been successfully ported to UQBAR. A new structure "s\_db.inc" has been added to the DEBERTH code, creating a new shared memory region for the DEBERTH variables. All of the test cases were completed and results were verified through hand calculations. The DEBERTH code is functional and will execute runs with and without damping for force and torque loads. Graphic results are included. The software is now ready for integration with the real time parallel process executive routine and simulation code.

## Appendix

TEST 1

# DI/DEBERTH Rel Pos Body1-2 vs. Time

dbresf1d0  
1 0 — 0 2

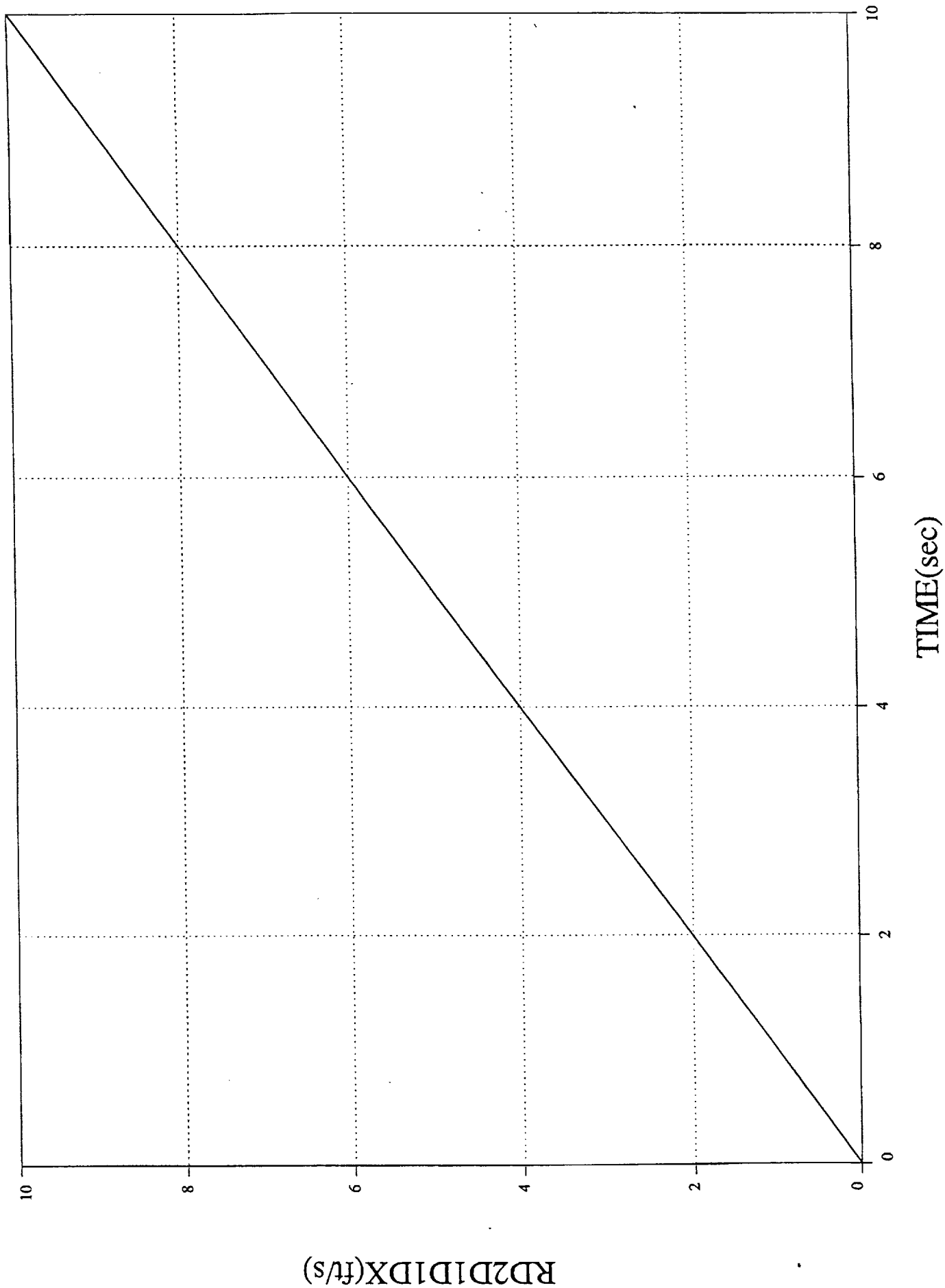




TEST 1

DI/DEBERTH Rel Vel Body1-2 vs. Time

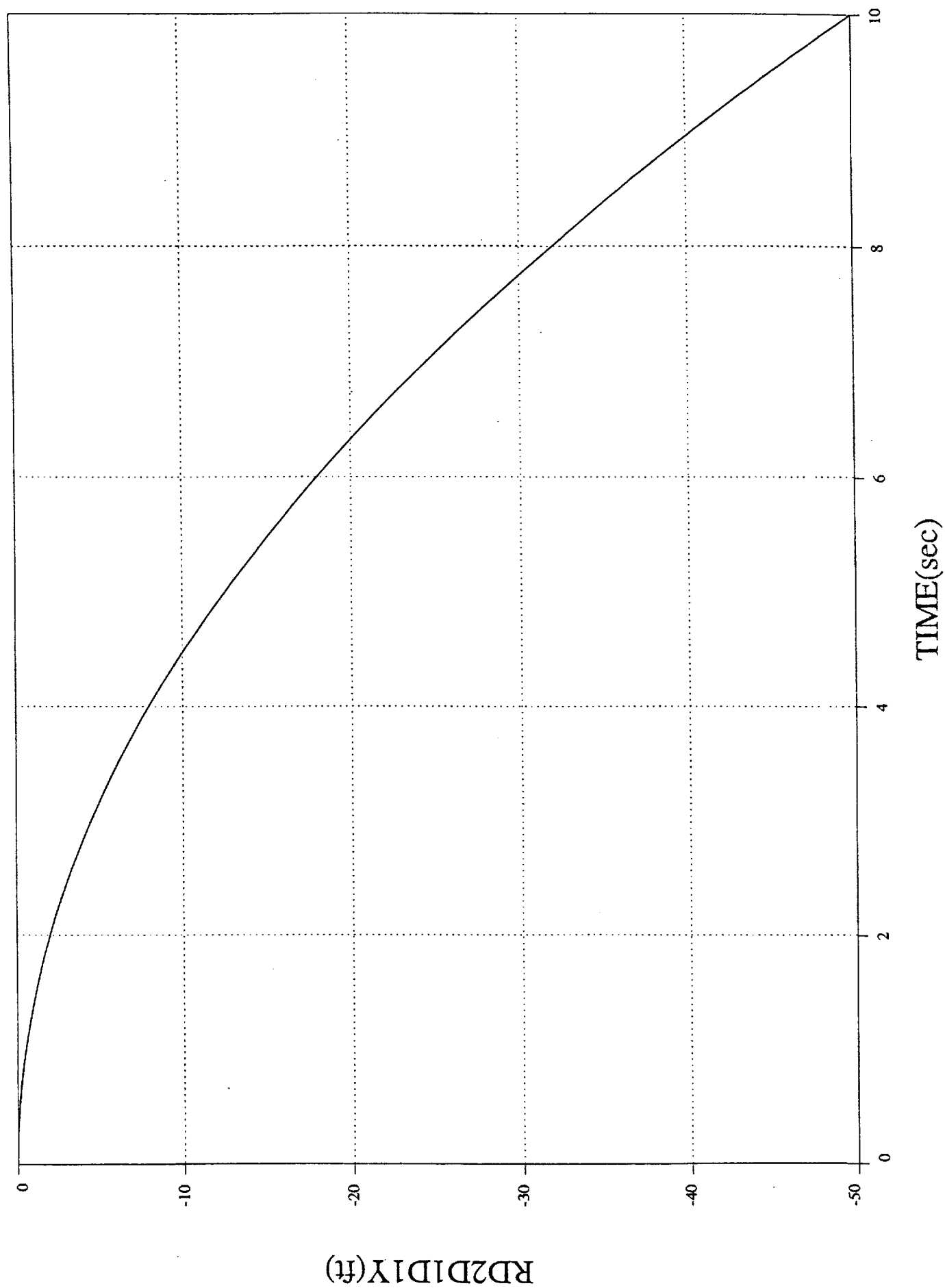
dbtestfid0  
1 0 — 0 27



TEST 2

DI/DEBERTH Rel Pos Body1-2 vs. Time

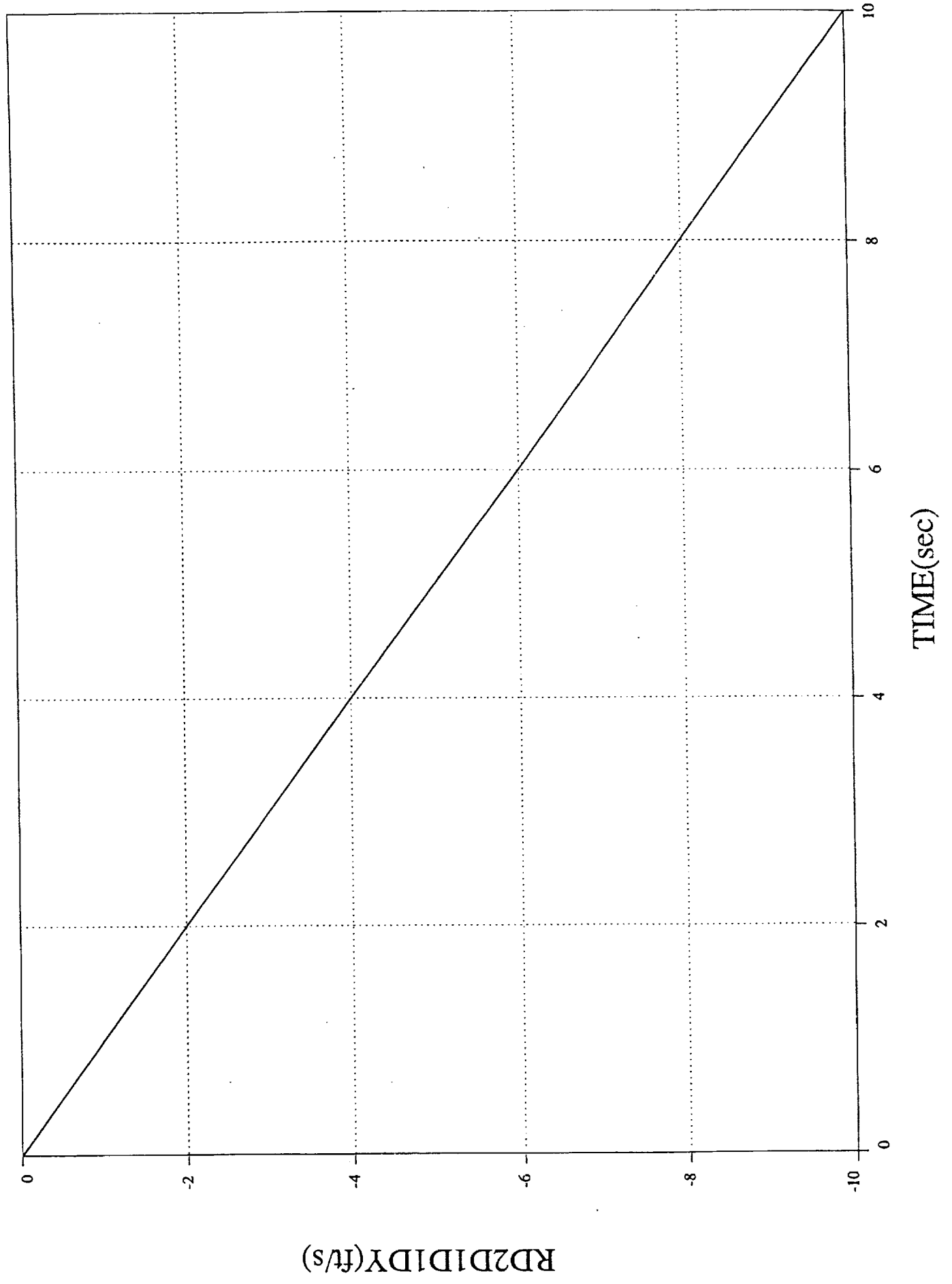
dbtestf2d0  
1 0 3



TEST 2

DI/DEBERTH Rel Vel Body1-2 vs. Time

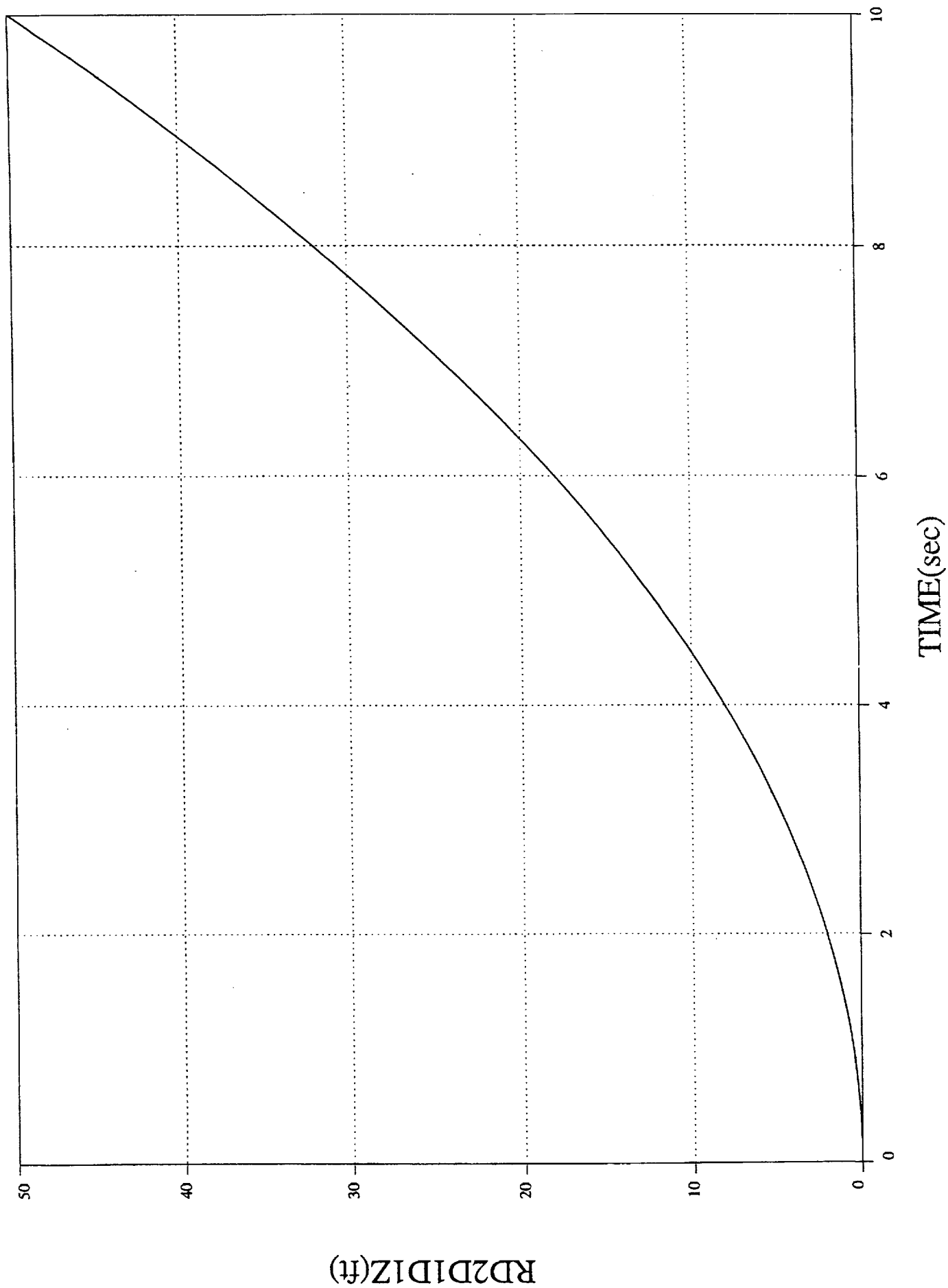
dbrestf2d0  
1 0 28



TEST 3

DI/DEBERTH Rel Pos Body1-2 vs. Time

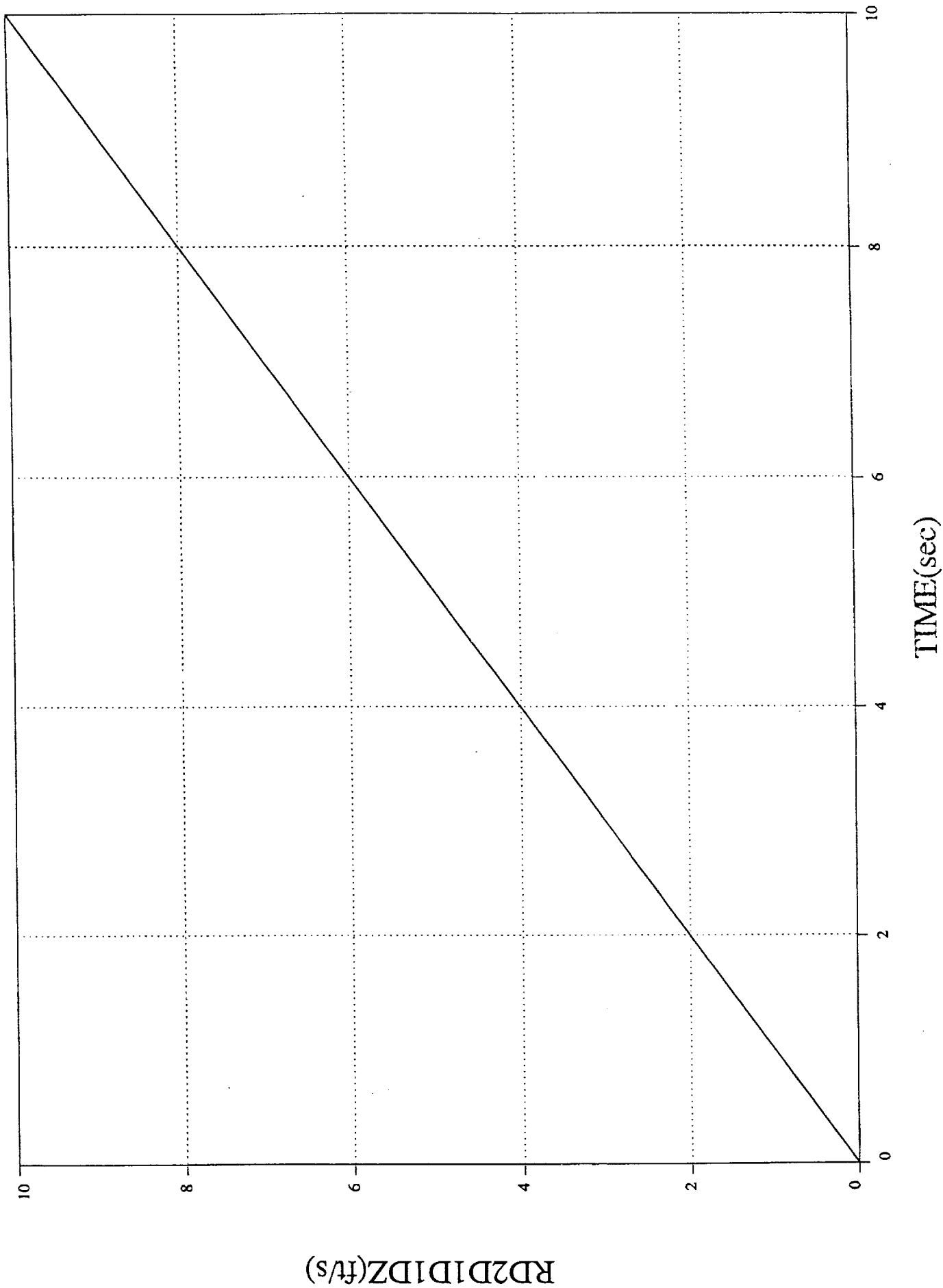
dbtestf3d0  
1 0 — 4



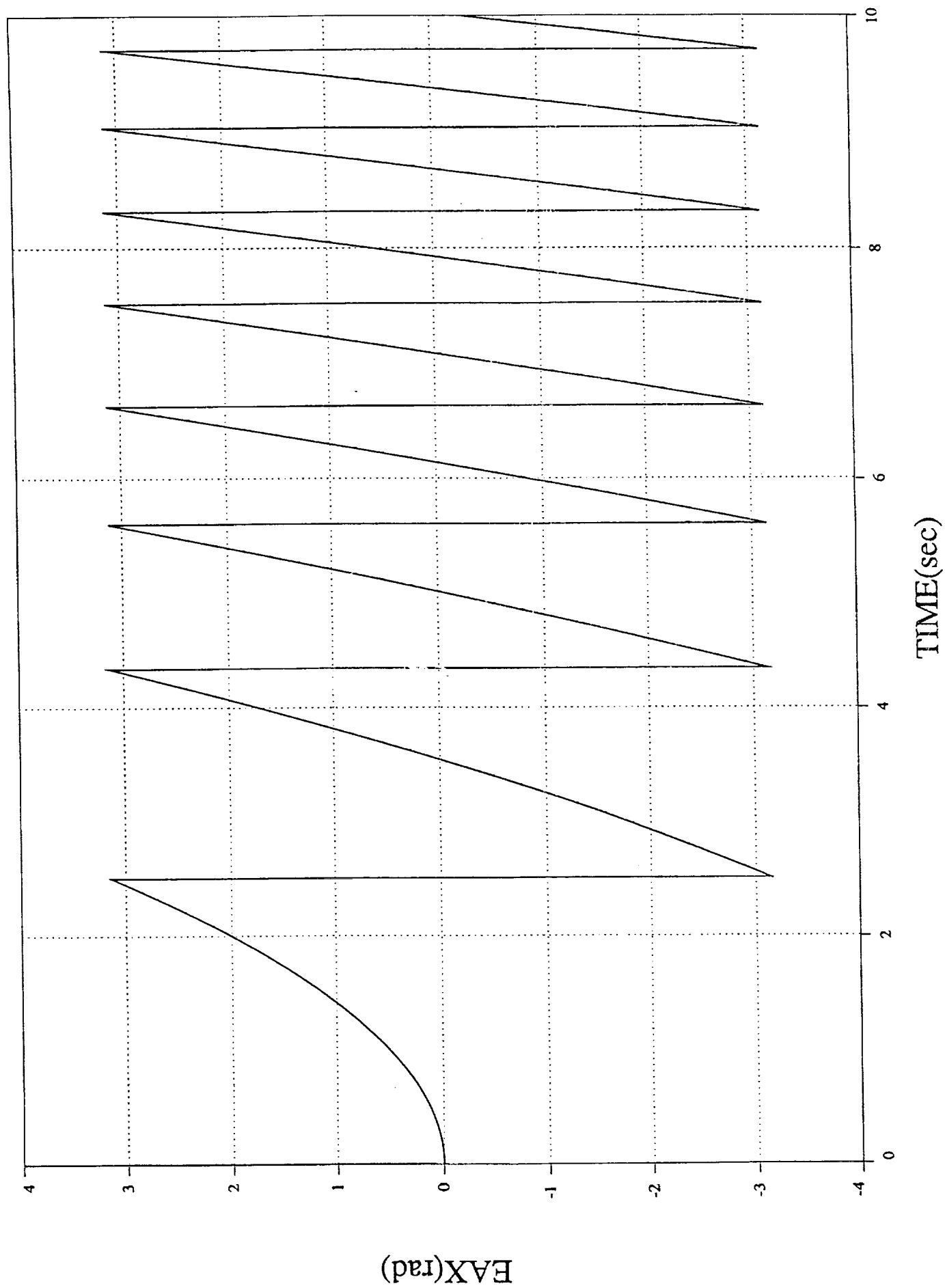
TEST 3

DI/DEBERTH Rel Vel Body1-2 vs. Time

dbtestf3d0  
1 0 — 0 29

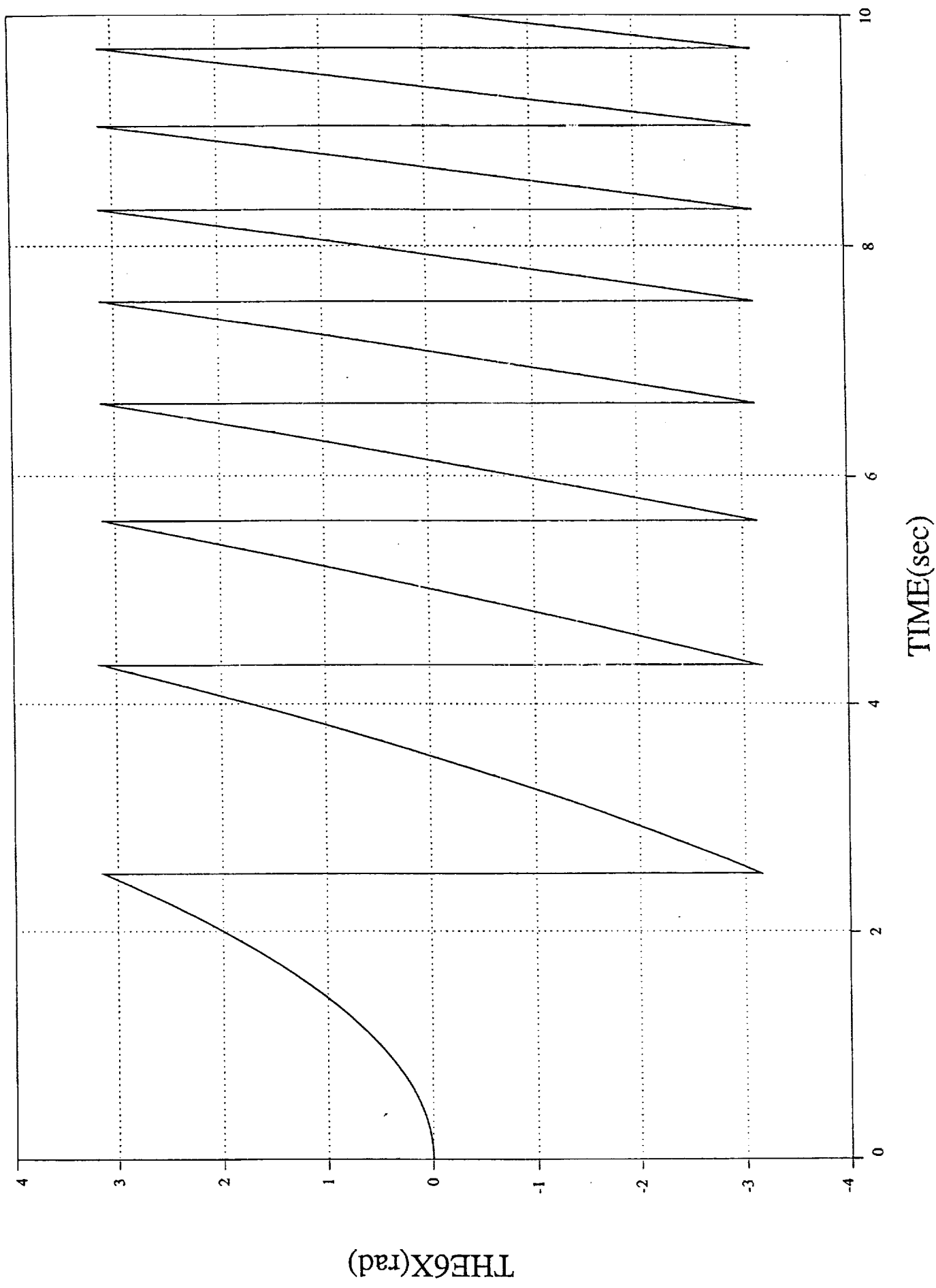


## DI/DEBERTH Euler Angle Z vs. Time

dbtest1d0t  
1 0 — 0 32

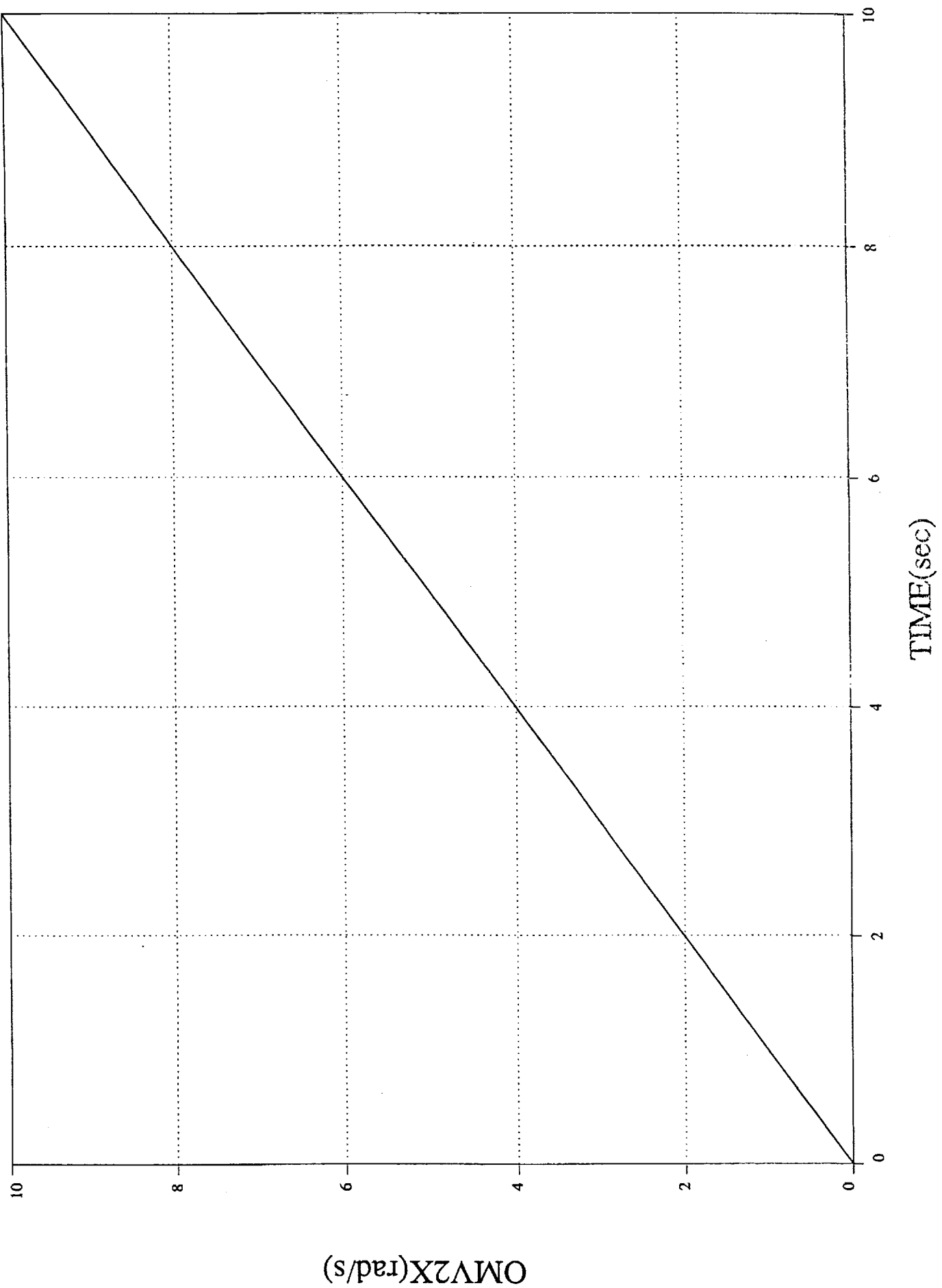
# DI/DEBERTH Rel Rotation vs. Time

dbtesttd0t  
1 0 — 24



DI/DEBERTH Angular Velocity vs. Time

dbtest11d0t  
1 0 — 5

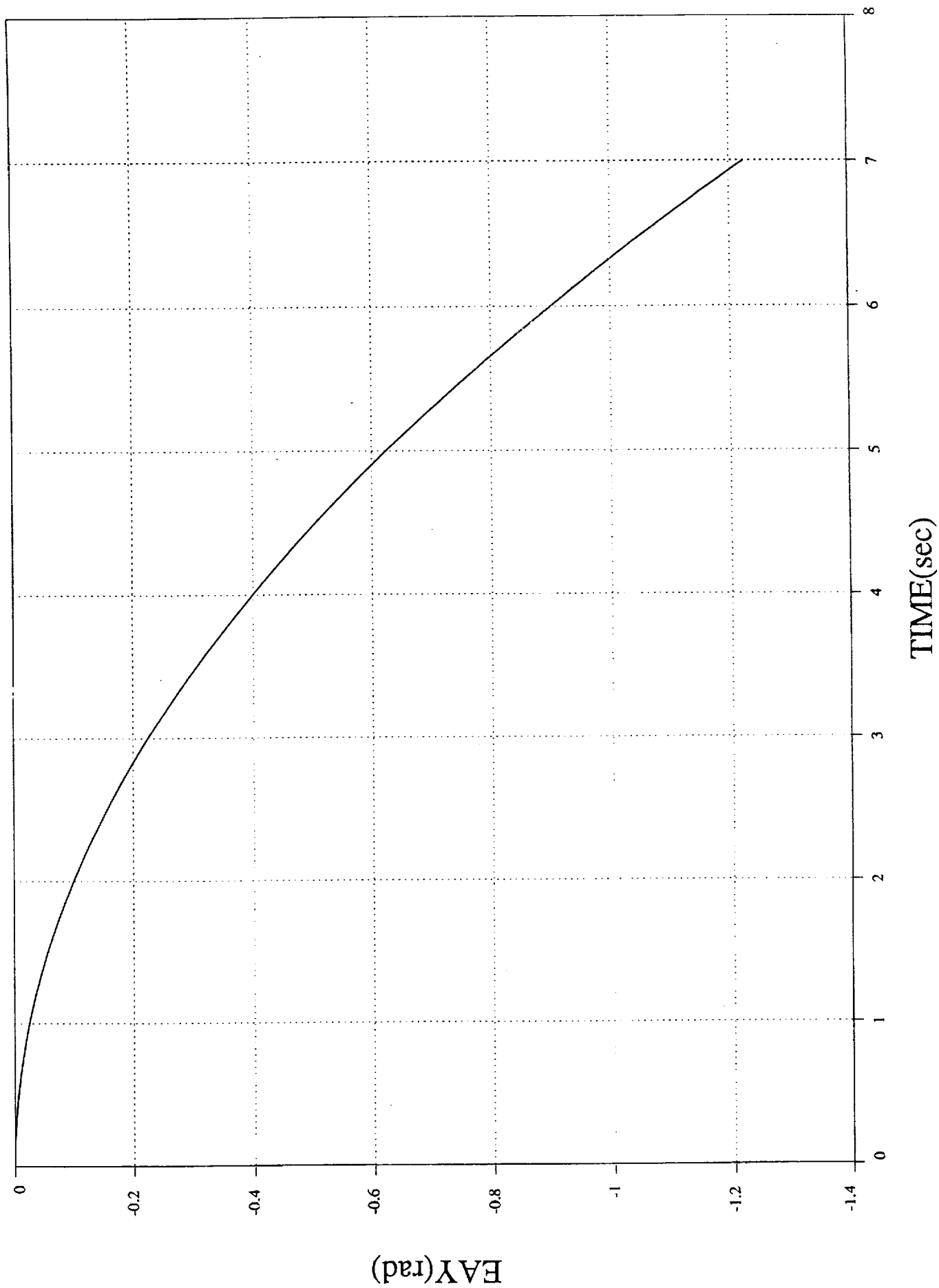




TEST 5

# DI/DEBERTH Euler Angle 2 vs. Time

10 — 31  
nogimble5

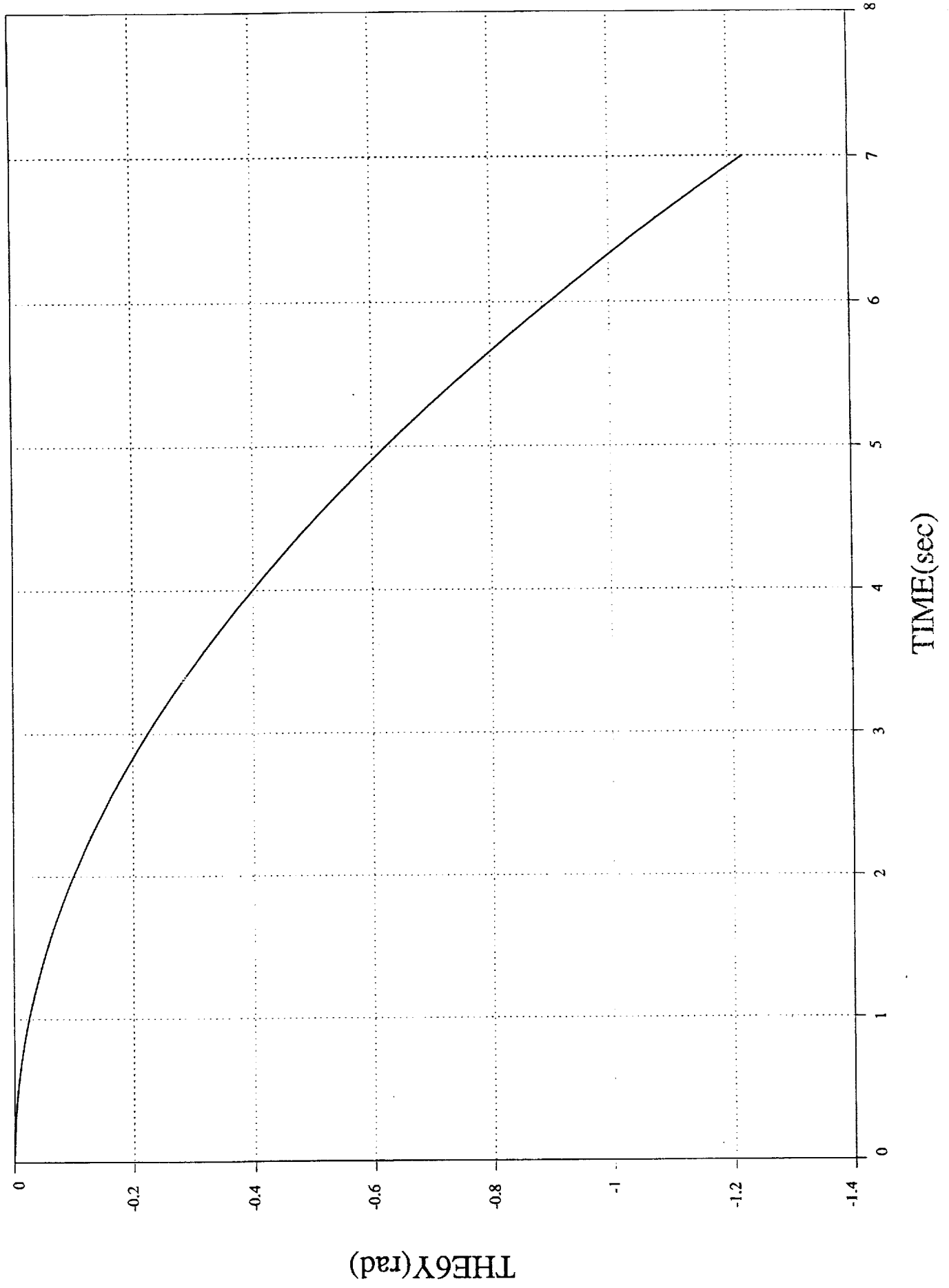


TEST5

# DI/DEBERTH Rel Rotation vs. Time

no gimble5

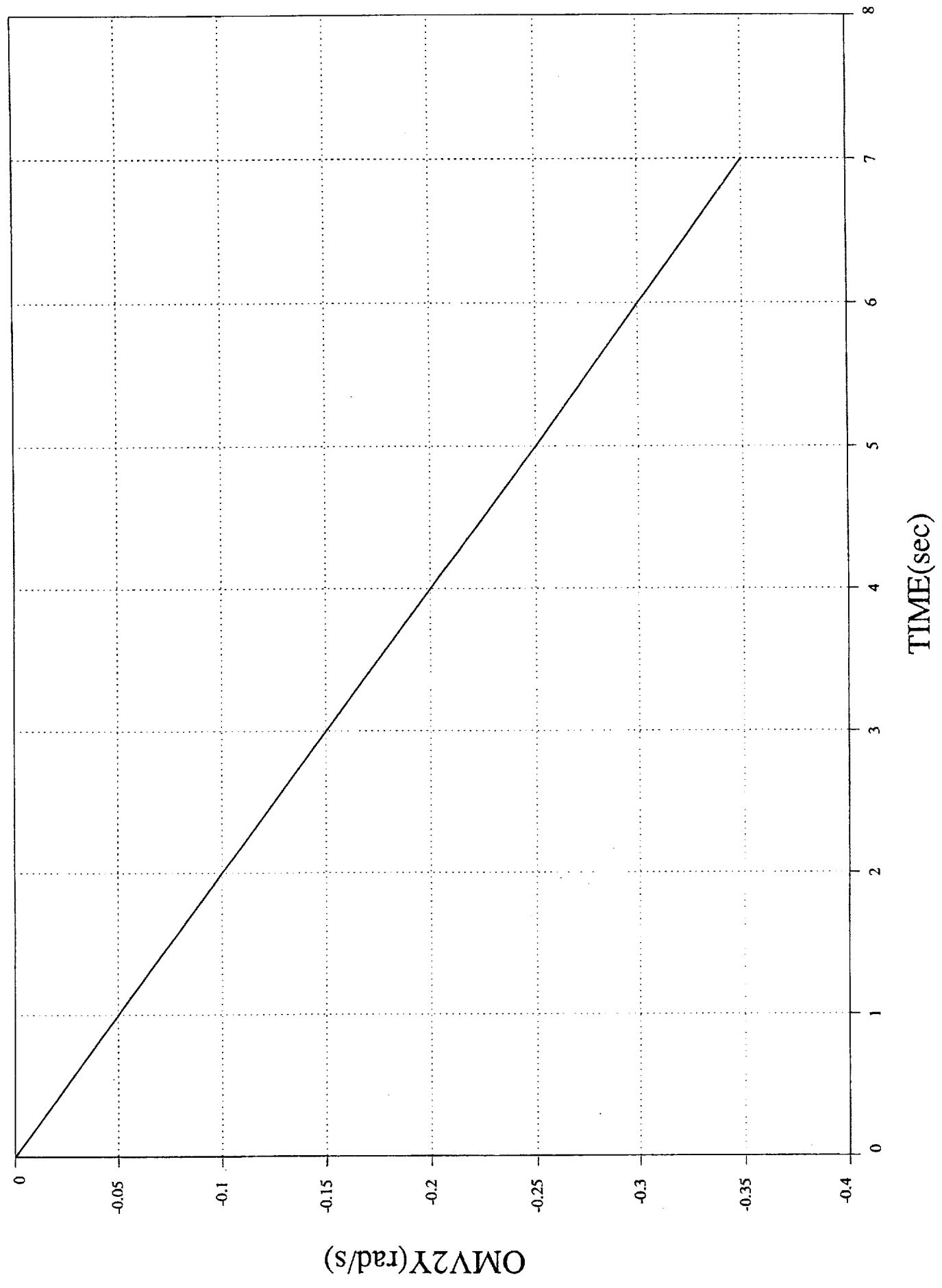
1 0 — 25



TEST 5

# DI/DEBERTH Angular Velocity vs. Time

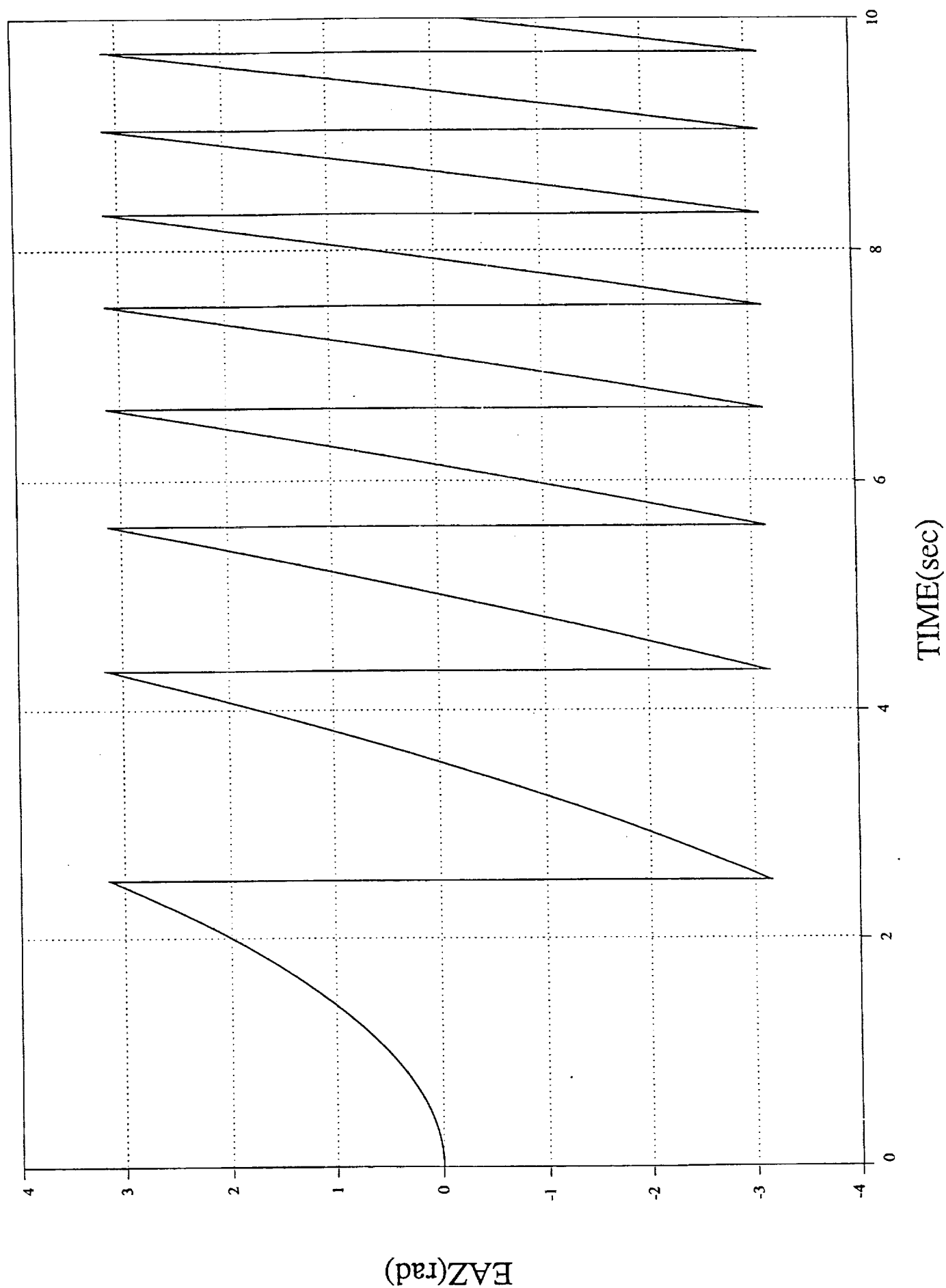
1 0 6  
nogramble5



TEST 6

# DI/DEBERTH Euler Angle Z vs. Time

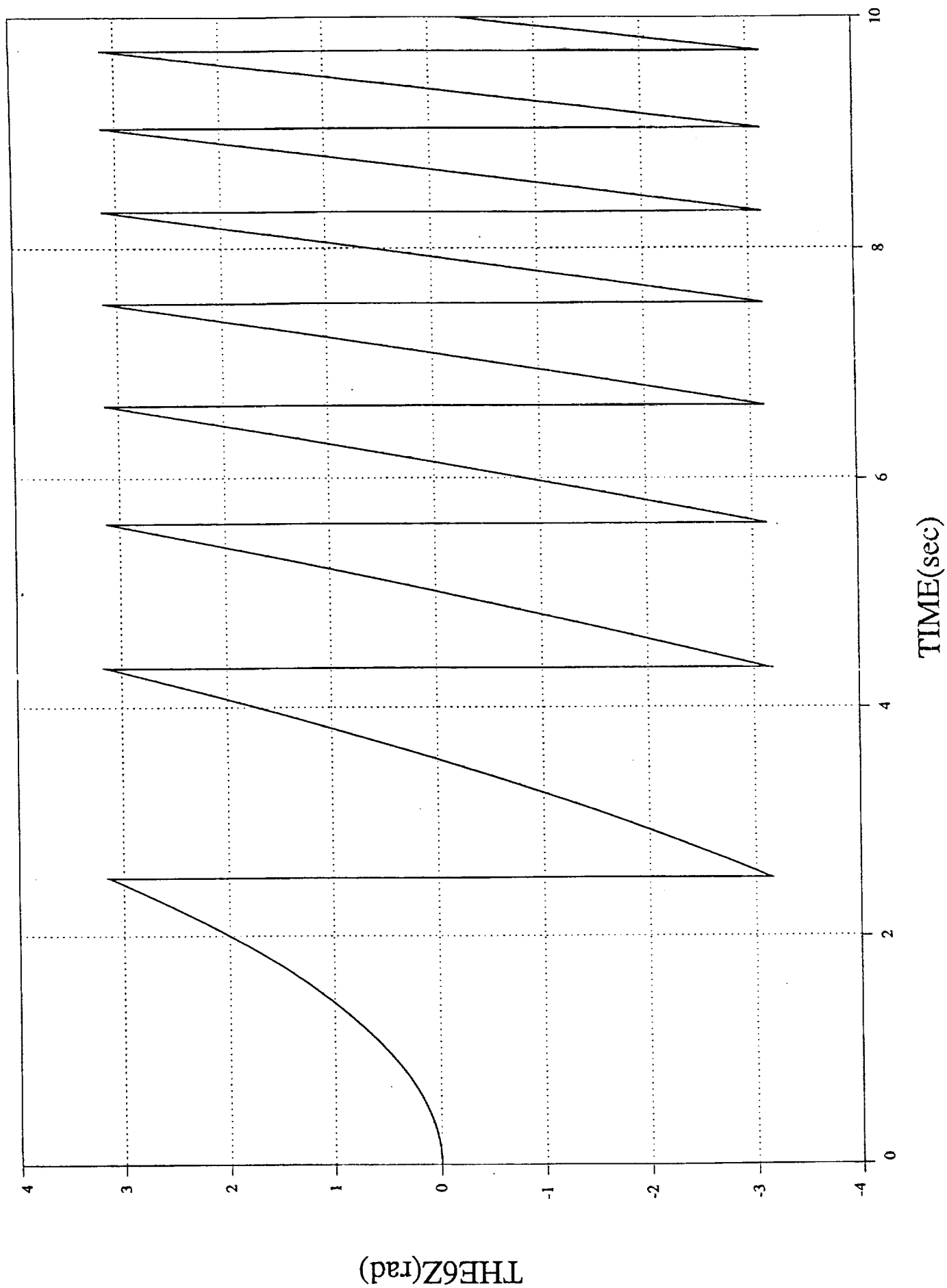
dbtest3d0t  
1 0 — 30



TESTC

# DI/DEBERTH Rel Rotation vs. Time

dbtest3d0t  
1 0 — 26

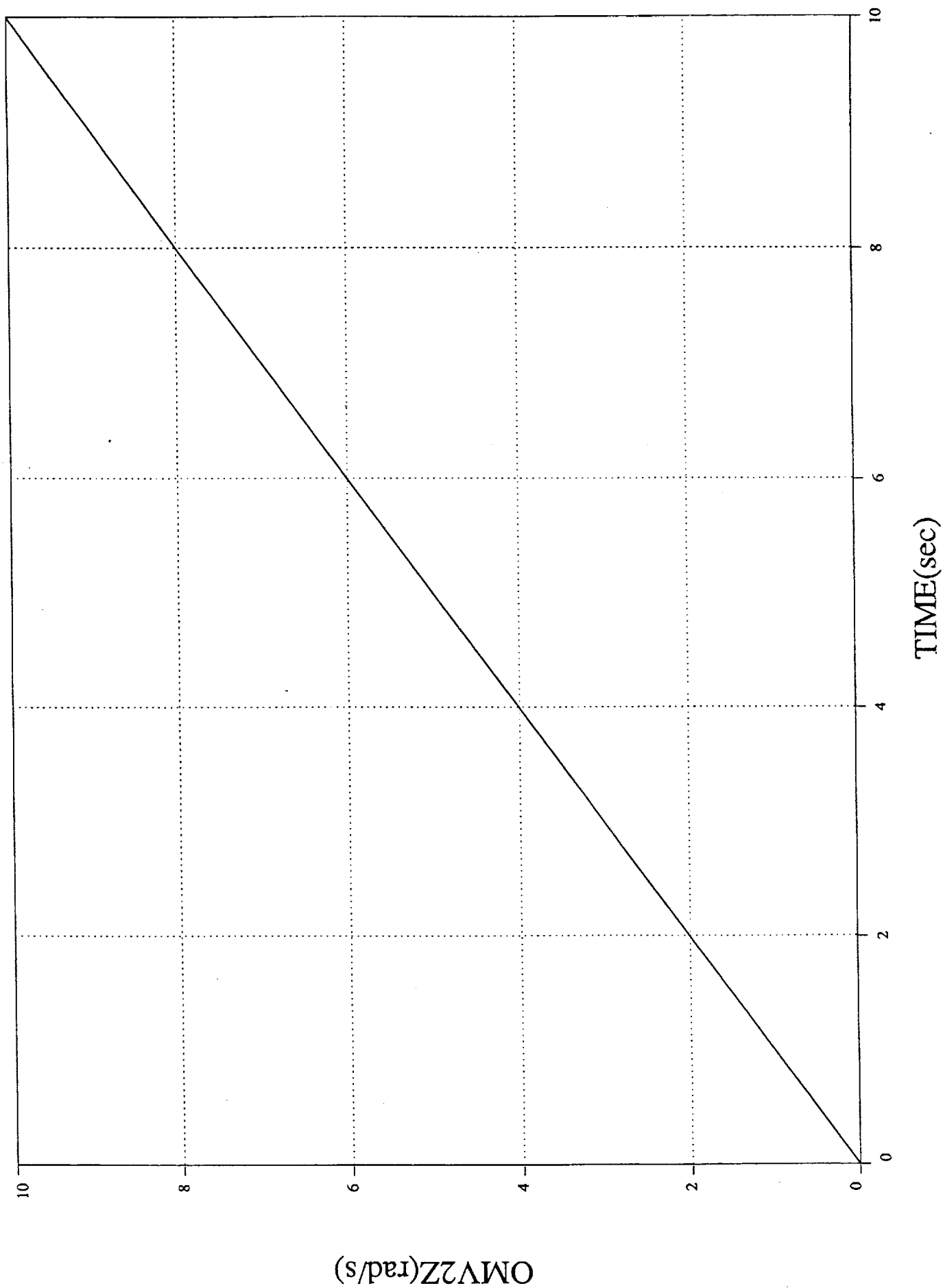


TEST 6

# DI/DEBERTH Angular Velocity vs. Time

dbtest3d0t

1 0 — 7



TEST 7

MANUAL VERIFICATION

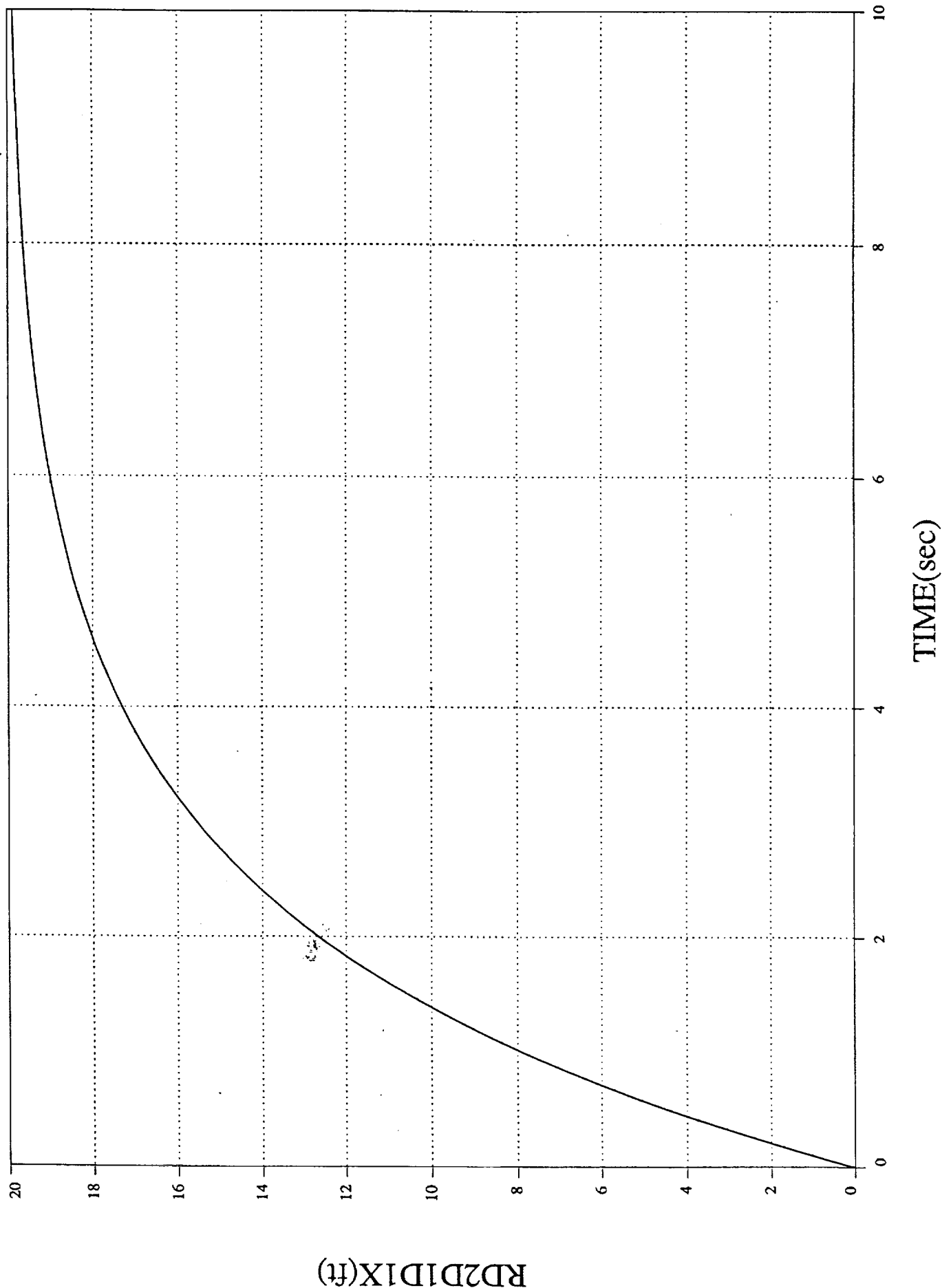
ON P.2

by DE.

DI/DEBERTH Rel Pos Body 1-2 vs. Time

dbstf0d50

1 0 — 2



MANUAL VERIFICATION

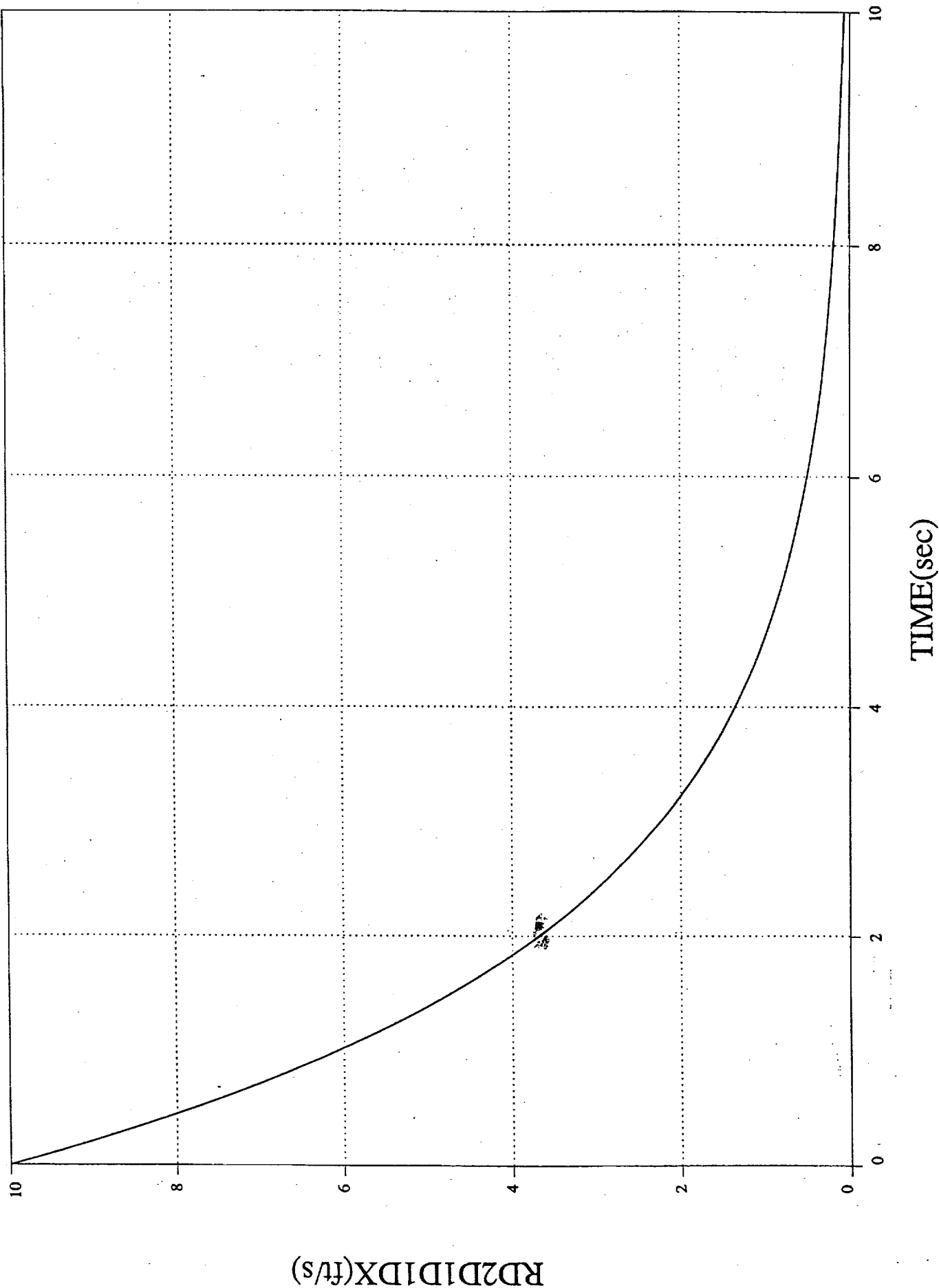
ON P.2

BY D.E.

# DI/DEBERTH Rel Vel Body1-2 vs. Time

dbtstf0d50

1 0 ——— 0 27





## Reference 10

DCI TM#012798-1  
2BODY Real-Time Simulation Integration and Validation  
January 27, 1998

## **Dynamic Concepts Technical Memorandum #012798-1**

To: bd Systems - Mr. Ronald Francis  
From: Dynamic Concepts - Dr. Patrick Tobbe and Mr. Jimmy Compton  
Subject: 2BODY Real-Time Simulation Integration and Validation  
Date: January 27, 1998

---

### **1. Introduction**

The real time simulation 2BODY is hosted on the SGI Challenge computer UQBAR and simulates two rigid bodies in contact. The Dynamic Concepts Technical Memorandum #011398-1 addresses recent code modifications which streamlined the code and test cases which verified the simulation running independent of the real time executive (twobod) developed for parallel operations. The purpose of this memorandum is to document the integration of the 2BODY software with the new parallel enabled executive and to present the results of validation runs conducted with the integrated simulation.

### **2. Software Integration**

The software integration process involved the development of drivers for the 2BODY simulation code, and the integration and debug of the 2BODY code with the new parallel processing executive software. In order to facilitate testing of the integrated simulation, several new features were incorporated into the 2BODY simulation.

#### **2.1 Integrated Software Structure**

Figure 1 shows the structure of the integrated 2BODY simulation software. The main routine "TWOBOD" is the new parallel processing executive which controls initialization, forks off parallel tasks, and performs task termination and clean-up. The new executive software is hosted on UQBAR under the directory: `usr/people/tobbe/di/source/twobod/`.

"DYNMAIN" is the driver developed by Dynamic Concepts for the two body contact dynamics simulation. The "DYNMAIN" driver and associated software are hosted on UQBAR under the directory: `usr/people/tobbe/di/source/2body/dyn_dock/`. "CTLMAN" is the driver developed by Dynamic Concepts for the software which interfaces with the panel control hardware. The "CTLMAN" driver and associated software are hosted on UQBAR under the directory: `usr/people/tobbe/di/source/2body/ctl_dock/`.

"ARC\_RUN" is a task which writes output data to files during the simulation run. "GFX" is a task which sends and receives data over an ethernet link to other SGI workstations running graphics display programs. The four tasks invoked by the executive are executed in a parallel manner on the SGI Challenge computer.

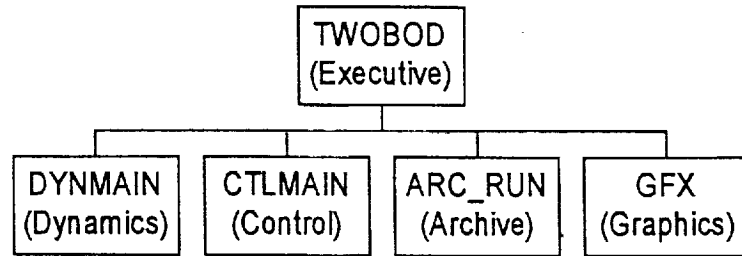


Figure 1 Integrated 2BODY Simulation Software Structure.

The makefile contained in the 2BODY run directory `usr/people/tobbe/di/exec/2body/` was updated to compile and link the new executive and the software associated with all supporting tasks.

All code modifications associated with the integration of the new executive were minor and mostly involved include file usage and the passing of structure pointers in the subroutine argument lists. Modifications to the real time executive software were clearly commented for future reference.

## 2.2 New Features

### 2.2.1 Software Only Mode

In order to test the integrated simulation independent of the facility hardware, a "software only" run mode was incorporated to circumvent the use of hardware components. Logic was added around subroutine calls which invoke hardware in order to check for the "software only" mode of operation. To define the mode of operation, the flag "SW\_MODE" was added to the structure defined in "s\_tb.inc". The "start.f" code was modified to prompt the user for the simulation operating mode (HW or SW\_ONLY). If the user selects the software only mode, the simulation prompts the user for a simulation stop time in seconds. The simulation stop time is contained by the variable "SW\_MODE\_TSTOP" which was added to the structure defined in "s\_tb.inc". In the case of a "software only" operating mode, logic was added to terminate the simulation run based on the value of "SW\_MODE\_TSTOP".

### 2.2.2 Spring Contact Force/Torque Model

In order to provide force/moment sensor outputs in the case of a "software only" operating mode, a spring contact force/torque model (CFM) was incorporated into the 2BODY simulation. The spring CFM for the 2BODY simulation was developed based on the spring model used for verification in the Table Contact Dynamics Simulation (TCDS). The source code for the 2BODY spring CFM is contained in the file `usr/people/tobbe/di/source/2body/dyn_dock/spring_cfm.f`. The associated makefile "2bdynmake" was modified to compile and link the spring CFM source code.

The spring CFM can model a linear spring (translational displacement), a torsional spring (rotational displacement), or output constant force/torque sensor measurements. The configuration of the spring CFM is governed by the parameters contained in the input data file `usr/people/tobbe/di/exec/2body/spring_cfm.dat`.

### 3. Integrated Simulation Validation Runs

Eight docking runs were conducted to validate the 2BODY simulation integrated with the new parallel enabled executive.

#### 3.1 Run Descriptions

Table 1 describes the rigid body mass properties of the two interacting bodies. Note that the mass and inertia of body one are much larger than that of body two in order to simulate an "infinite" mass body interacting with a small body.

Table 2 describes the eight test cases for the integrated 2BODY simulation. Test cases one through six involve the application of constant uniaxial forces and torques, applied independently, about each of the three cardinal axes. Test case seven utilizes the linear CFM and chosen initial conditions on body two in order to produce contact via a translational displacement. Test case eight employs the torsional CFM and initial conditions on body two to produce contact through rotational displacements.

Table 1 2BODY Rigid Body Mass Properties.

| Body       | Mass (slugs) | Inertia (ft-lb-sec <sup>2</sup> ) |
|------------|--------------|-----------------------------------|
| 1 - Target | $10^{25}$    | $10^{30}$ (diagonal)              |
| 2 - Chaser | 10           | 10 (diagonal)                     |

Table 2 2BODY Validation Test Cases.

| Case | Load Type            | Load Axis | Load                  | ICs  |
|------|----------------------|-----------|-----------------------|--|
| 1    | Force                | X         | 10 lb                 | 0  |
| 2    | Force                | Y         | 10 lb                 | 0  |
| 3    | Force                | Z         | 10 lb                 | 0  |
| 4    | Torque               | X         | 10 ft-lb              | 0  |
| 5    | Torque               | Y         | 10 ft-lb              | 0  |
| 6    | Torque               | Z         | 10 ft-lb              | 0  |
| 7    | Linear Spring CFM    | Z         | Translational Contact | Position [0 0 3.0833] ft<br>Velocity [0 0 -0.5] ft/s |
| 8    | Torsional Spring CFM | Z         | Rotational Contact    | Euler Angles [0 0 0] deg<br>Ang Rate [0 0 0.5] deg/s |

### 3.2 Test Results

Since body one is a stationary "infinite" mass, the motion of body two is the relative motion between body one and body two. The single axis translational and rotational accelerations of body two,  $a_2$  and  $\alpha_2$ , are given by

$$F_2 = m_2 a_2$$

$$T_2 = I_2 \alpha_2$$

For constant accelerations, the translational position and velocity of body two may be computed from

$$R_2 = v_0 t + a_2 t^2 / 2$$

$$d(R_2)/dt = v_0 + a_2 t$$

and the rotational position and velocity may be computed from

$$\theta_2 = \omega_0 t + \alpha_2 t^2 / 2$$

$$d(\theta_2)/dt = \omega_0 + \alpha_2 t$$

For zero initial conditions, the translational position and velocity of body two are given by

$$R_2 = a_2 t^2 / 2$$

$$d(R_2)/dt = a_2 t$$

and the rotational position and angular velocity are given by

$$\theta_2 = \alpha_2 t^2 / 2$$

$$d(\theta_2)/dt = \alpha_2 t$$

#### 3.2.1 Constant Force/Torque Results

The constant force and torque test cases were designed to simplify the results verification process by using mass, inertia, and load parameters which greatly simplify the equations of motion. Test cases one through six were set up such that  $F_2=10$  lb,  $m_2=10$  slugs,  $T_2=10$  ft-lb, and  $I_2=10$  ft-lb-sec<sup>2</sup>. These parameters produce  $a_2$  and  $\alpha_2$  values of 1 ft/s<sup>2</sup> and 1 rad/s<sup>2</sup>, respectively. Therefore, a graph of the translational position due to a constant force should start at zero and increase parabolically to 50 ft at 10 seconds. The corresponding translational velocity should start at zero and increase linearly to 10 ft/s at 10 seconds.

For the rotational cases, the Euler angles should start at zero and increase parabolically to 2 rads (114.6 deg) at 2 seconds, and the angular velocity should start at zero and increase linearly to 2 rad/s (114.6 deg/s) at 2 seconds. Graphs of the test results are included in Appendix A.

### 3.2.2 Spring Contact Force/Torque Results

Test case seven was designed such that body two will contact the linear spring at 5 seconds traveling with a translational velocity of -0.5 ft/s in the body two Z-axis. Due to mounting offsets and the uncompressed length of the spring, contact should occur when body two's docking port is 0.5833 ft from the docking port of body one. During the contact period, the force applied to body two by the spring model should be proportional to the deflection of the linear spring model as given by

$$F = K_x \delta_x$$

where  $K_x$  represents the linear spring stiffness and  $\delta_x$  represents the translational deflection of the spring. Since the linear spring is a conservative system, body two should rebound with a velocity magnitude that is equal to the velocity magnitude prior to contact. The case seven graphs in Appendix A confirm this behavior.

Test case eight was designed such that body two will contact the torsional spring at 2 seconds traveling with a angular velocity of 0.5 deg/s about the body two Z-axis. The torsional spring model is a dual leaf spring configurable with a deadband region about the reference position. In this test case, the angular deadband region is  $\pm 1$  deg. During the contact period, the torque applied to body two by the spring model should be proportional to the deflection of the dual leaf spring model as given by

$$T = 2 R_{arm} K_r \delta_r$$

where  $K_r$  represents the torsional spring stiffness,  $\delta_r$  represents the rotational deflection of the spring, and  $R_{arm}$  represents the moment arm associated with a single side of the leaf spring. Since the torsional spring is a conservative system, body two should rebound from contact with an angular velocity magnitude that is equal to the velocity magnitude prior to contact. Also, angular momentum should be conserved during the contact/rebound process. The case eight graphs in Appendix A bear out this behavior.

## 4. Conclusions

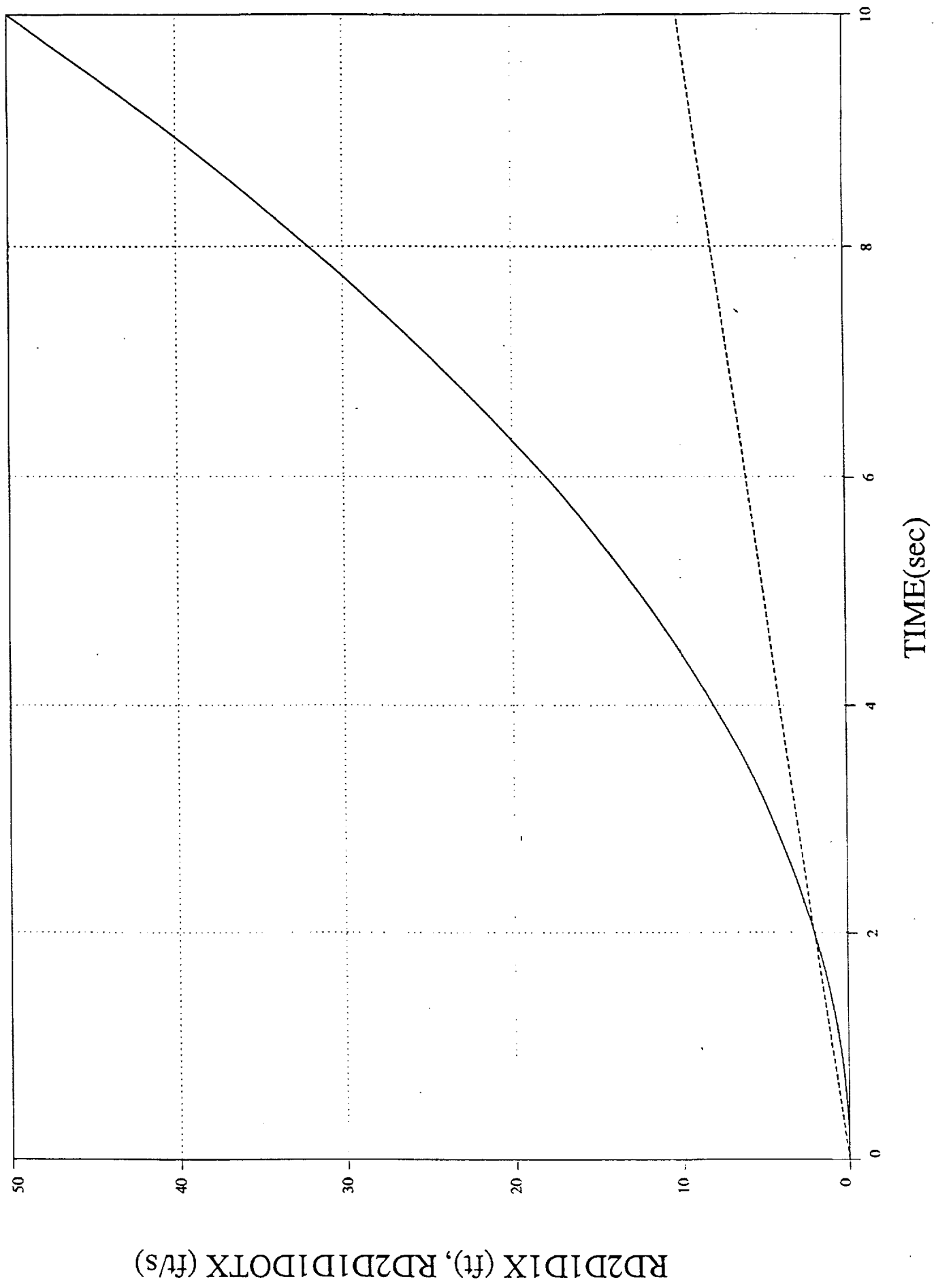
Drivers for the two body contact dynamics simulation were developed and successfully integrated with the new parallel processing executive. Several new features were implemented in the integrated 2BODY simulation to facilitate validation testing. These new features include a "software only" mode of operation and a spring contact force/torque model for use in place of hardware. Eight test cases were conducted and the test results were verified through hand calculations in order to validate the integrated 2BODY simulation software.

## Appendix A

### Simulation Results

Case1: 2Body Constant Uniaxial X-Force

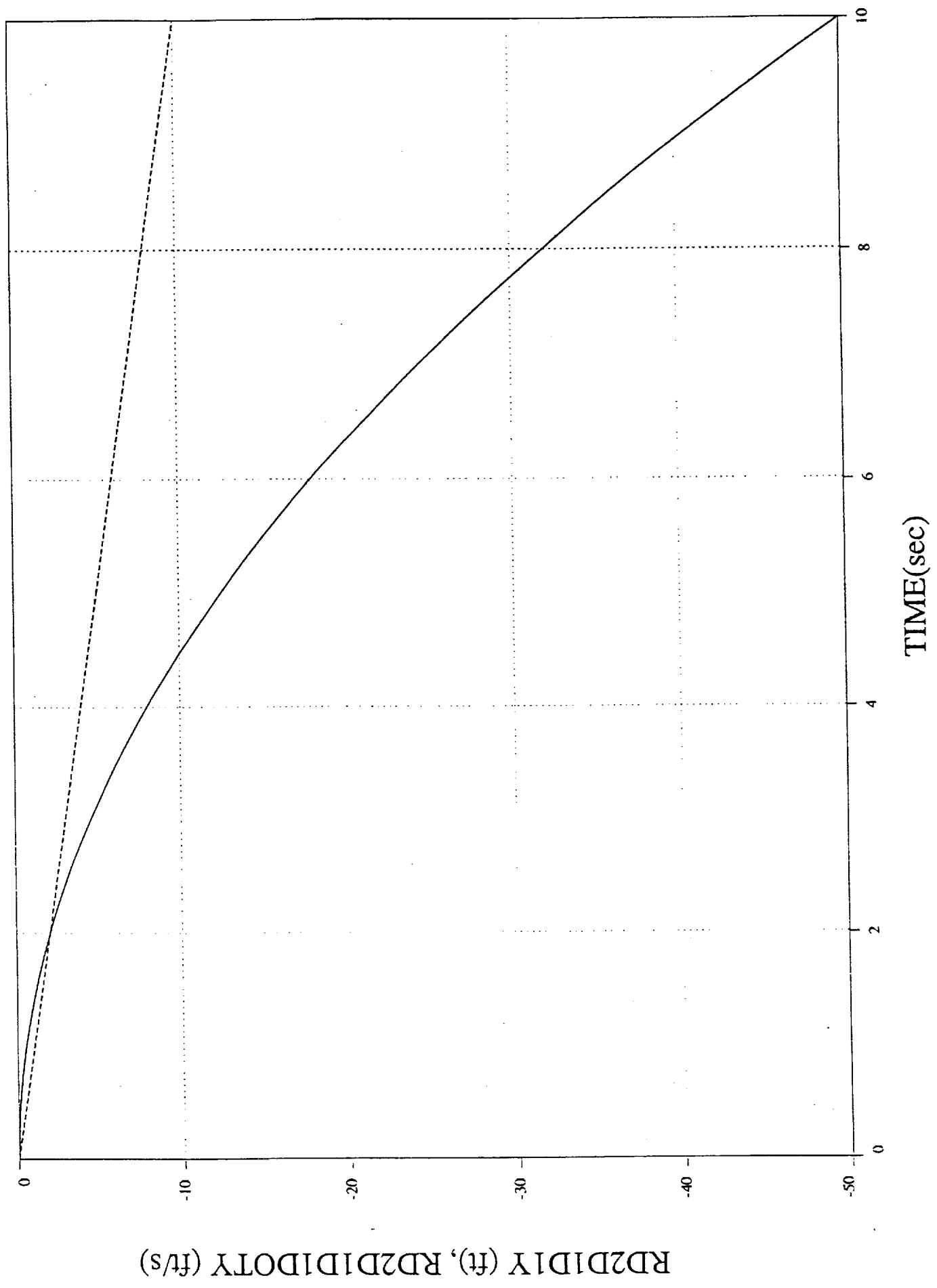
f1esid1 1 0  
f1esid1 2 0  
f1esid1 49 1 Δ





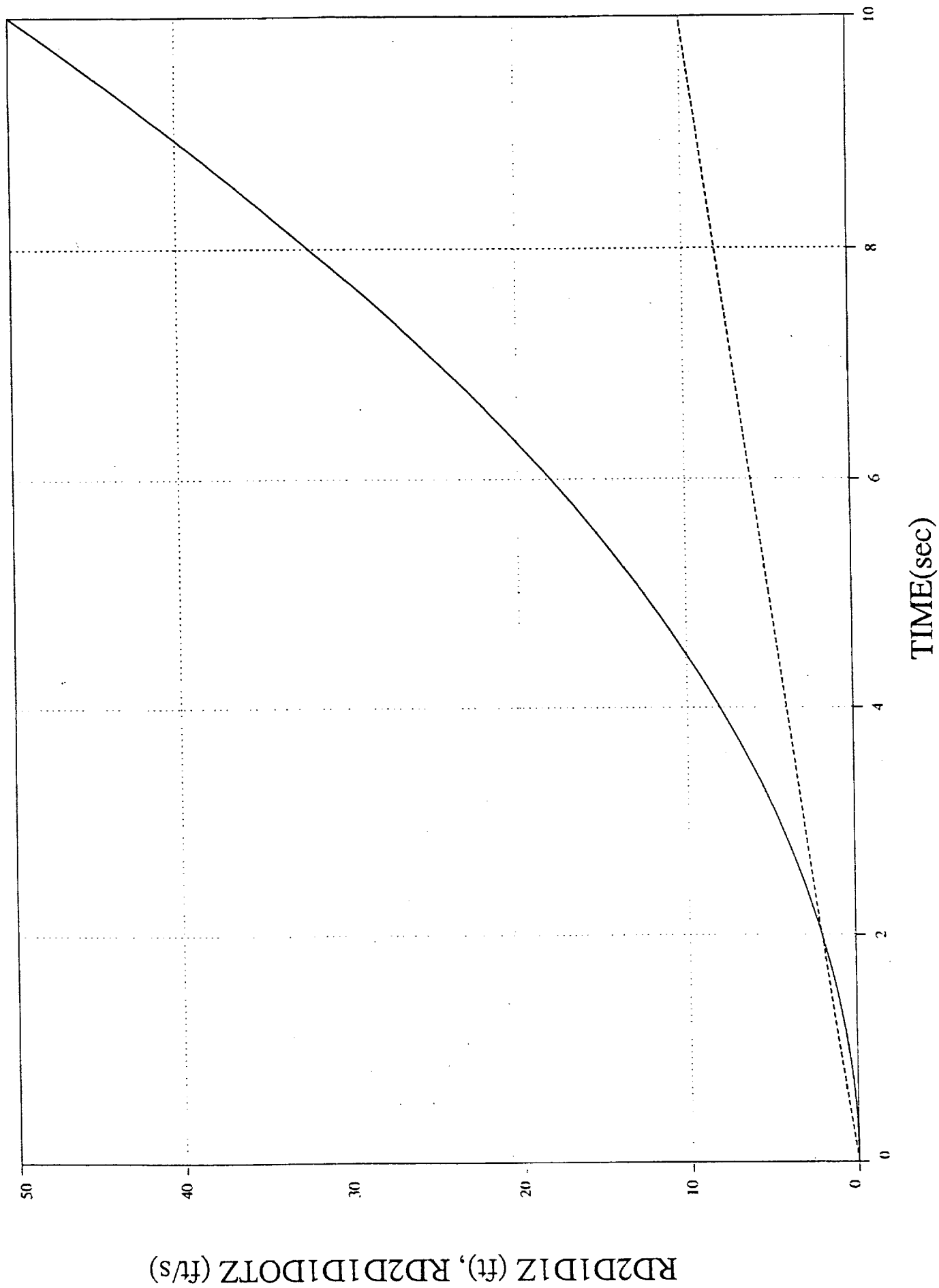
# Case2: 2Body Constant Uniaxial Y-Force

fresid1 1 3 1 Δ 50



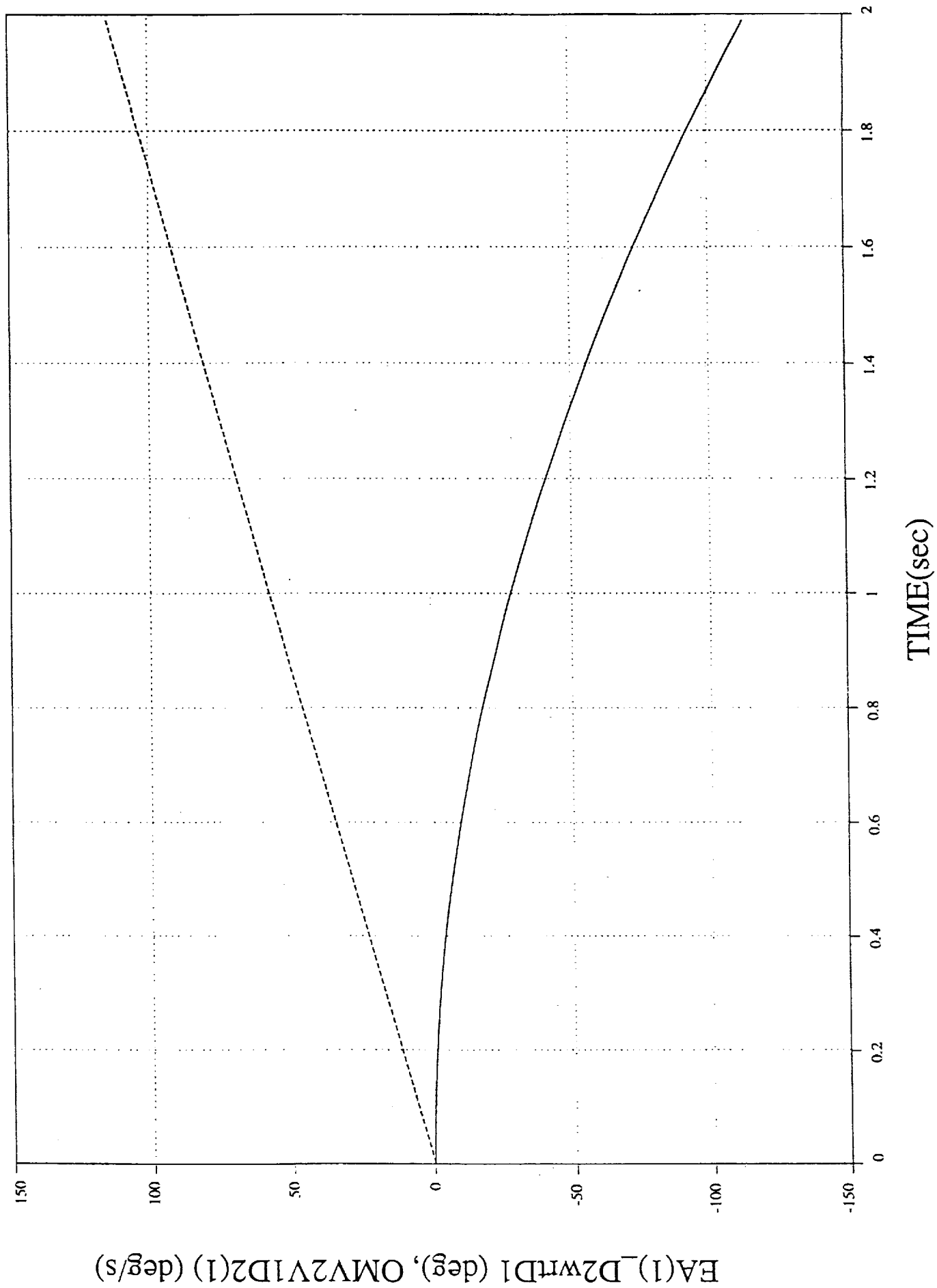
# Case3: 2Body Constant Uniaxial Z-Force

f1esd1 1 4 51  
f1esd1 1 4 51



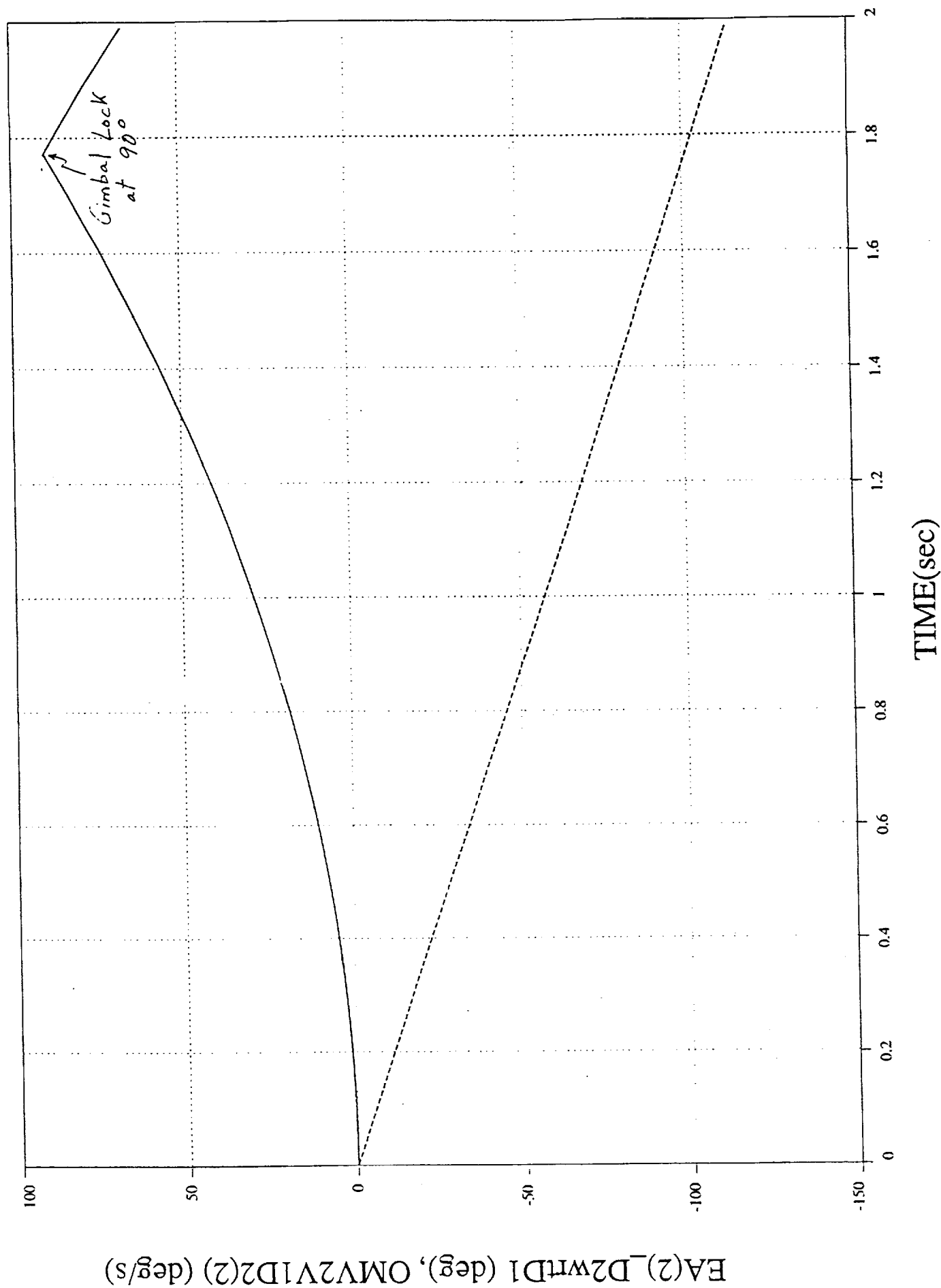
# Case 4: 2Body Constant Uniaxial X-Torque

Legend:  
 1 ○ — 5  
 1 △ --- 8



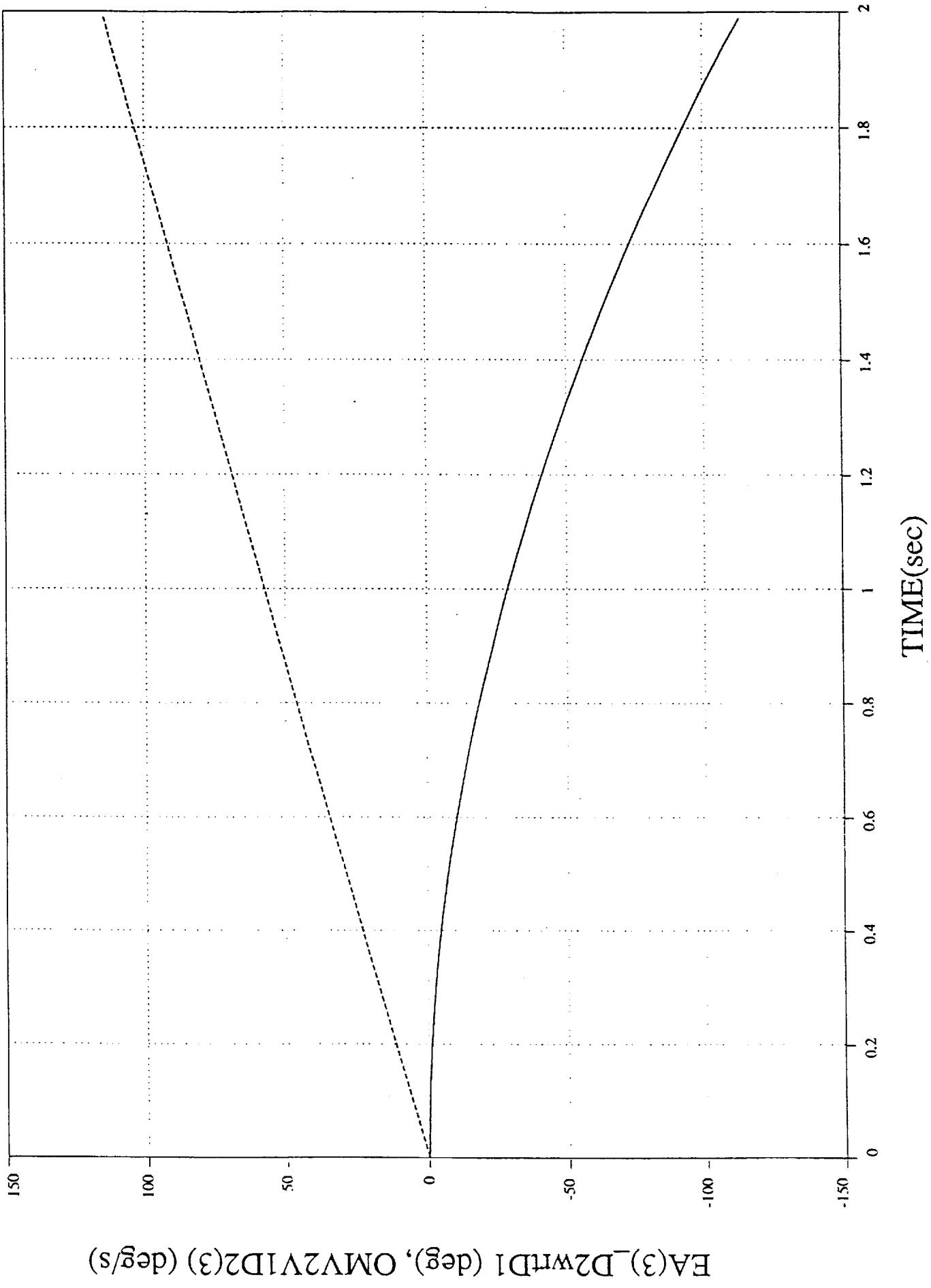
# Case 5: 2Body Constant Uniaxial Y-Torque

11espyd1 1 0 6 9



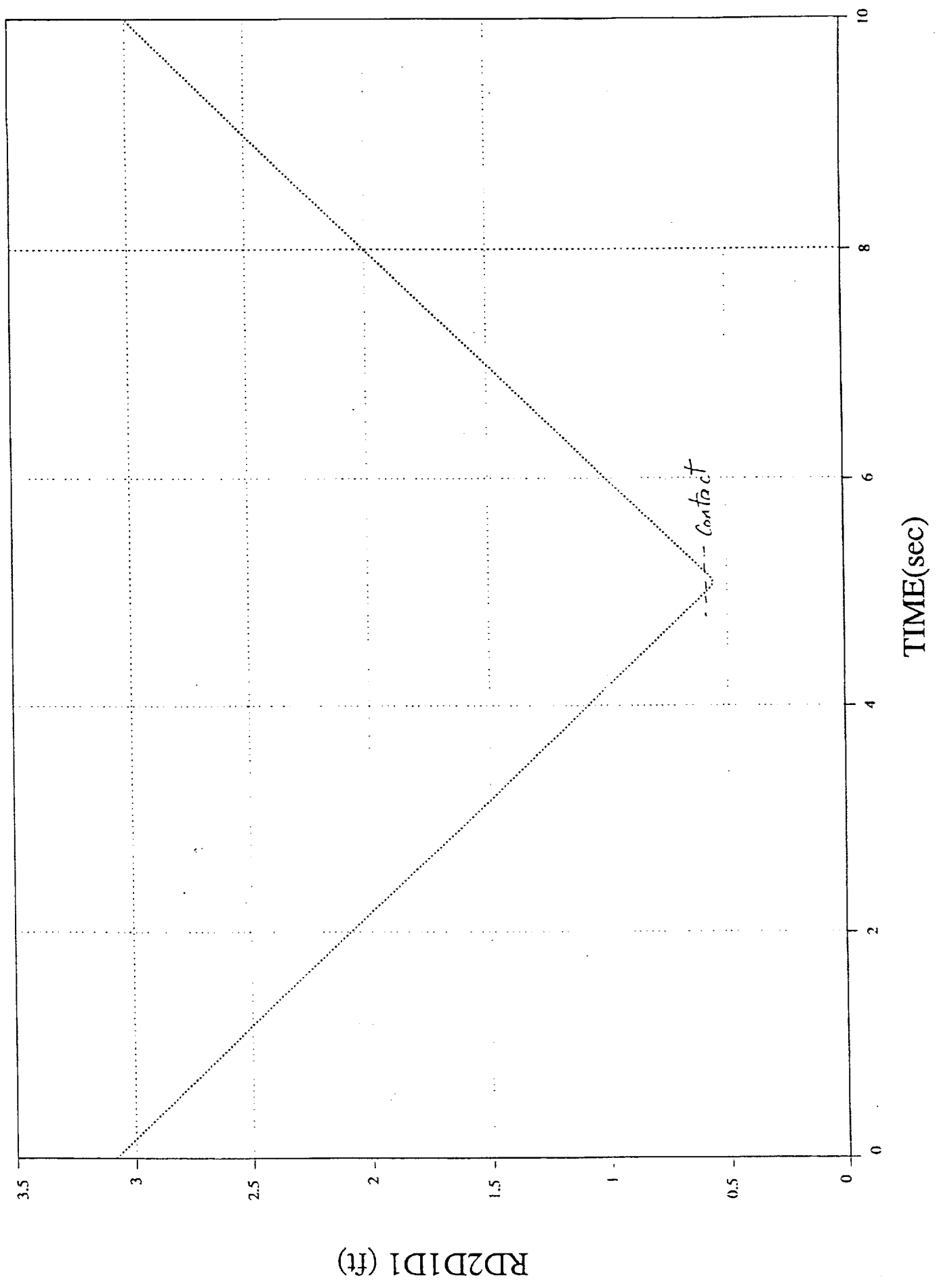
# Case 6: 2Body Constant Uniaxial Z-Torque

Legend:  
1  $\bigcirc$  —  $\bigcirc$  7  
1  $\Delta$  - - -  $\Delta$  10

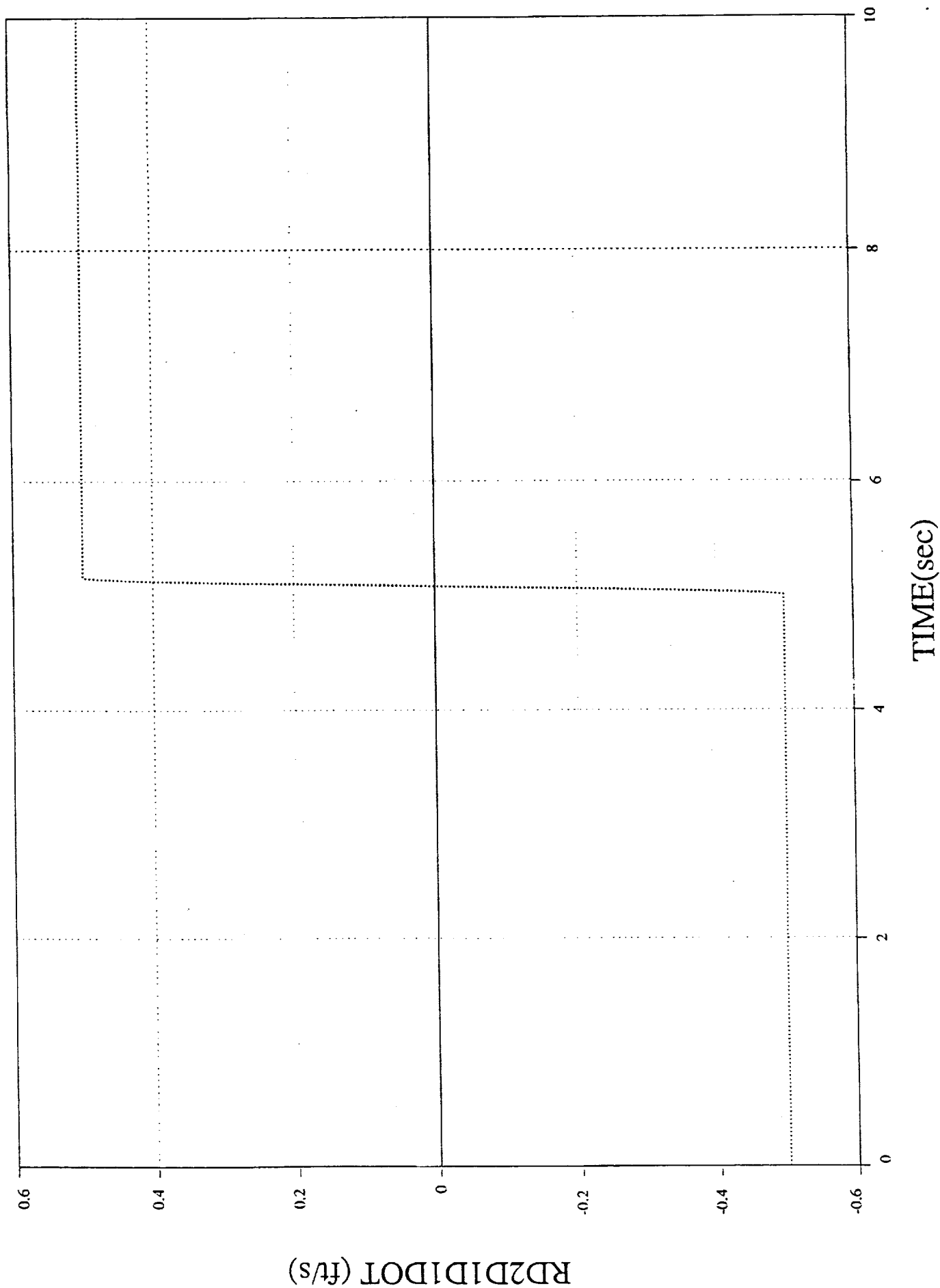


$f_{spring10d1}$   $\bigcirc$  —  $\bigcirc$  2       $f_{spring10d1}$   $\Delta$  - - -  $\Delta$  3       $f_{spring10d1}$   $\square$  .....  $\square$  4

Case 7: 2Body Linear Spring CFM



fspring10d1 Case 7: 2Body Linear Spring CFM<sup>spring10d1</sup>  
1 0 49 1 Δ 50 1 □ 51



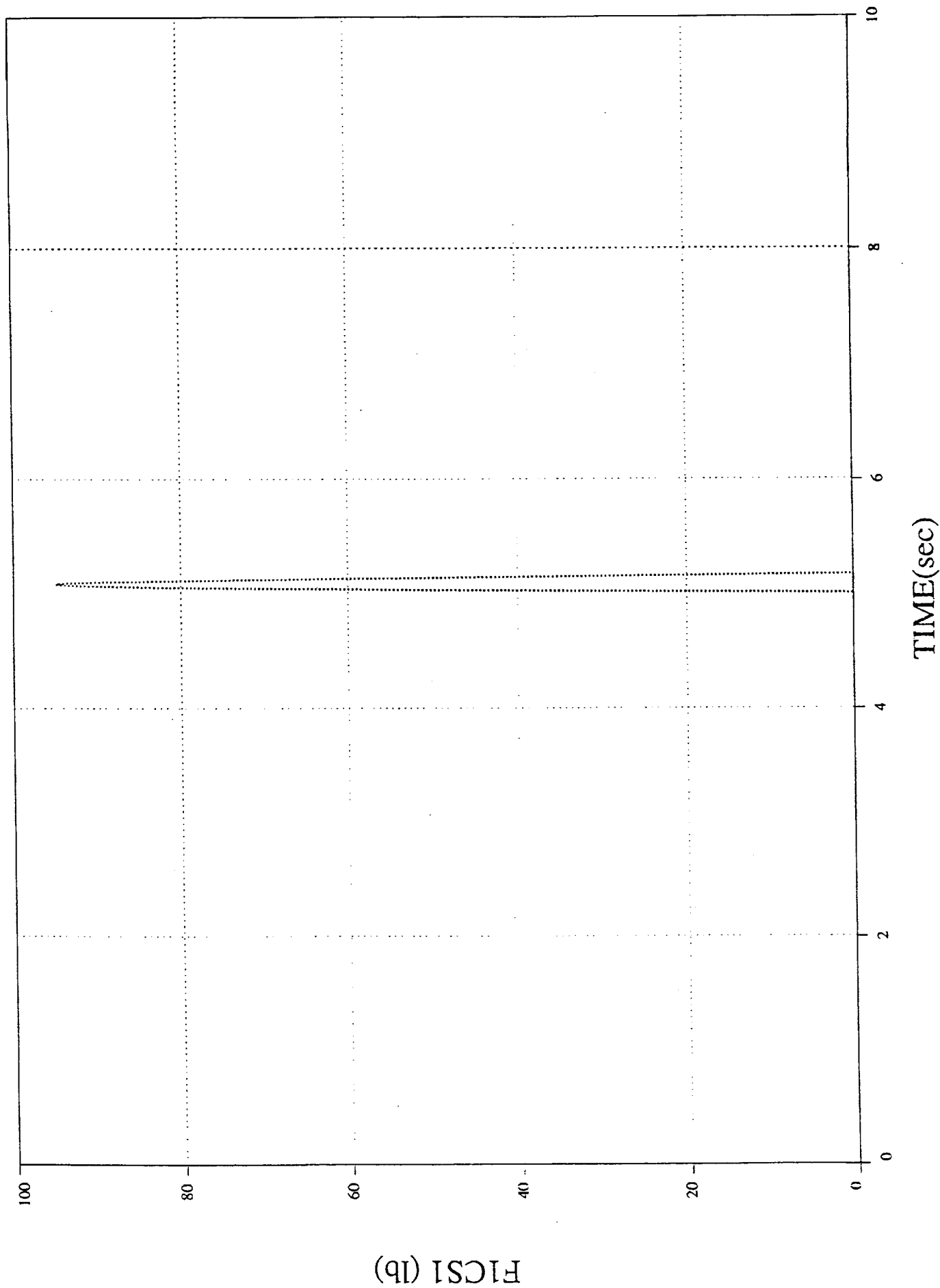
Case 7: 2Body Linear Spring CFM<sup>spring 10d1</sup>

fspring 10d1

1 ○ —○ 37

1 △ -----△ 38

1 □ .....□ 39





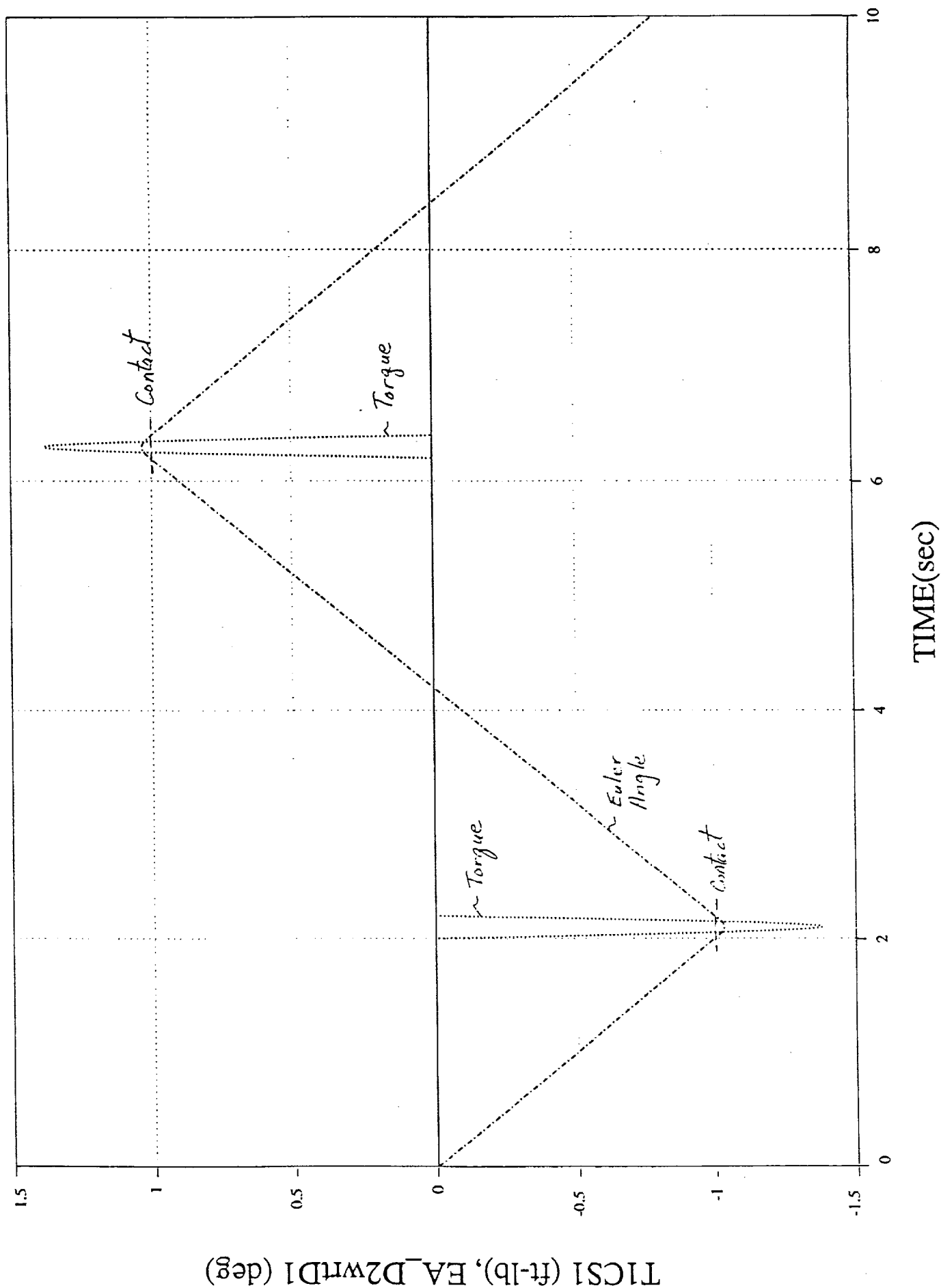
Case 8: 2Body Torsional Spring CFMfig10d1

1 (P01 Gujds)

Fig 10d

Spoliansi

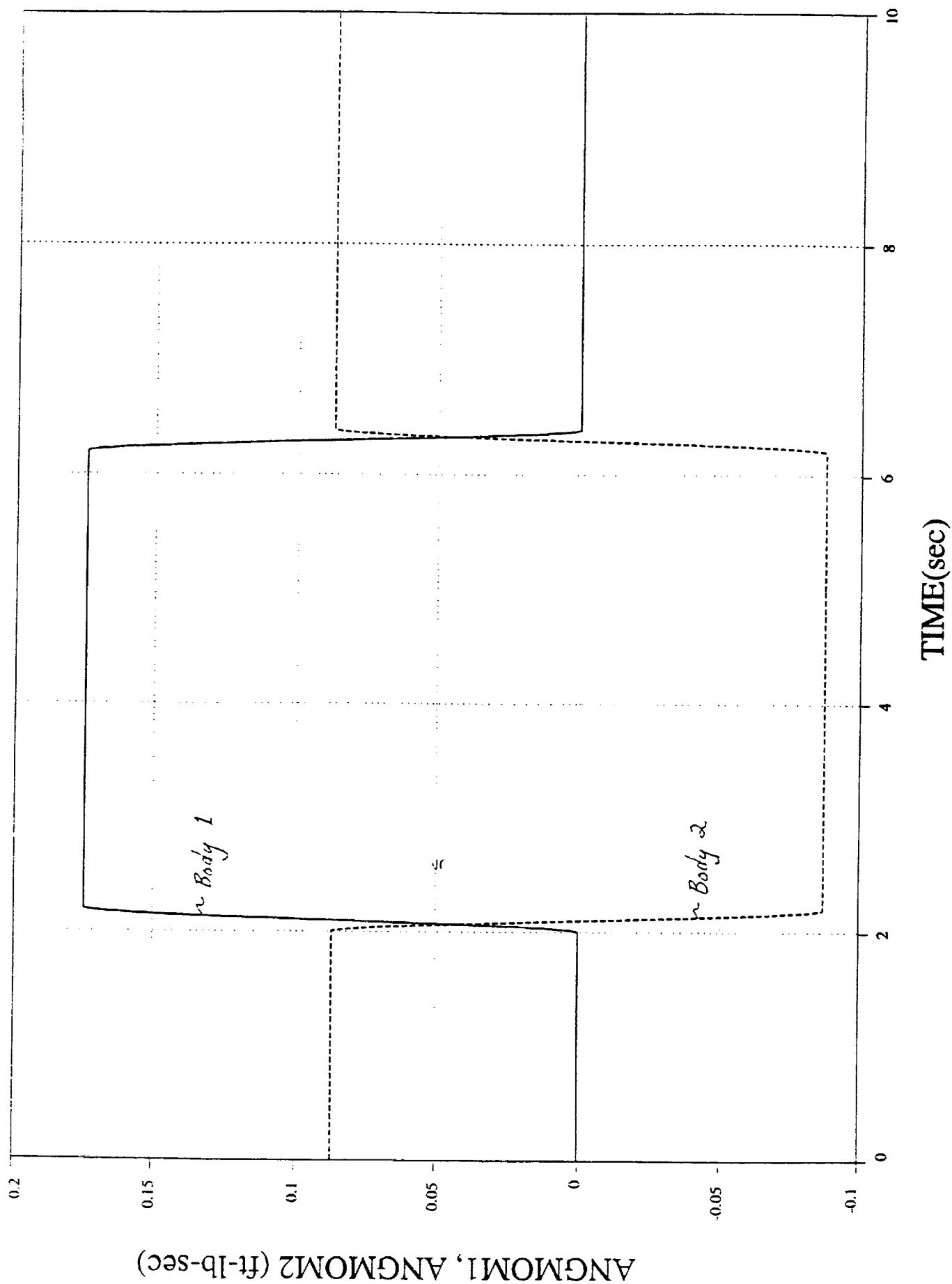
Isoprime 10x11



# Case 8: 2Body Torsional Spring CFM

tspring10d1  
1 Δ-----Δ 26

tspring10d1  
1 ○-----○ 13



## Reference 11

DCI TM#052398-1  
6DOF Facility Hardware / Software Integration  
May 23, 1998

## **Dynamic Concepts Technical Memorandum**

**#052398-1**

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts - Dr. Patrick Tobbe  
Subject : 6DOF Facility Hardware / Software Integration

### **Introduction**

This memorandum describes the results of the hardware / software integration tasks performed at the MSFC 6DOF facility. These activities were necessary to meet the requirements of integrated CBM test 275. The hardware / software integration process consists of a series of communications checks between the 6DOF hardware and the 2BODY and ROCKET simulations. These tests were followed by simple hardware in the loop simulations using springs as the test articles. Lastly, a set of ISS berths using the CBM hardware were performed with ROCKET and compared to previous test results.

### **Software Only Results**

As described in earlier memorandums, there are two real time simulation programs used at the 6DOF facility. 2BODY is a twelve degree of freedom simulation of two rigid bodies used to evaluate docking mechanisms. ROCKET is a real time berthing simulation utilizing the SRMS model. The software only test descriptions which follow were performed with both 2BODY and ROCKET.

Initially, a series of communication checks were executed between the 6DOF control panel and the simulation software. These tests consisted of verification of simulation start and stop commands, air table and force limit freeze commands, and initial simulation and table position commands. Next, a set of "rope pulls" were carried out between the force/moment sensor and the simulation program. In these cases, an external force and moment were applied to the load cell and the response of the math model was monitored from the control panel. A single axis force or moment was applied, sequentially in all six axes, to the sensor and the resultant direction of table commands were used to verify not only communications between the math model and the sensor, but also the orientation of the coordinate frames which link the facility to the model.

The deberth mode of operation was also exercised in the initial condition protect mode and simulation termination mode. In the simulation termination mode, execution of the model was stopped through the panel. The operator then entered deberth mode and drove the system with a series of artificial forces or thruster firings. The response of the system was then evaluated. The initial condition protect mode was verified by hitting the force/moment sensor with an external load while the table was ramping to a set of initial conditions defined by the math model. Upon seeing the load from the sensor, the model immediately transitioned into deberth mode and reacted accordingly.

Finally, the man-in-the-loop mode of operation for ROCKET was tested in a software only mode. Communications between the joy sticks and the SGI host UQBAR were tested and verified by MSFC. Upon validation of A/D software, several berthing approaches were performed by Mark Slone of MSFC and the resulting response of the RMS recorded. This data was analyzed to confirm the direction and duration of the commands and resultant arm motion.

### Hardware in the Loop Simulation Results

To verify hardware in the loop operation of the 6DOF facility, compensation tests performed with the previous version of 2BODY were repeated. CBM berths simulated with the earlier version of ROCKET were also approximated using three latches and a simulation cycle time of two milliseconds.

Translational compensation tests were run with 2BODY and the coil spring. These tests were a subset of runs made on 6/27/95 and used a cycle time of 2 milliseconds. Masses of 103.8, 524.9, and 1070.6 slugs were given various initial velocities and coasted into the spring in the single degree of freedom simulation. Previous compensation values were used for the tests and the exit velocities recorded. The test results presented in Tables 1 through 3 show excellent agreement between the current version of 2BODY on the SGI and the previous version of 2BODY on the Alliant system. Therefore, the same compensation curves for translation were retained in the new version of ROCKET.

| Run | Initial Velocity<br>in/sec | Compensation<br>Coefficient | Exit Velocity<br>in/sec 4/20/98 | Exit Velocity<br>in/sec 6/27/95 |
|-----|----------------------------|-----------------------------|---------------------------------|---------------------------------|
| 1   | -0.250                     | .065                        | 0.267                           | 0.251                           |
| 2   | -1.000                     | .053                        | 1.094                           | 1.037                           |
| 3   | -3.000                     | .050                        | 3.170                           | 3.080                           |

Table 1 : Translational Compensation Results - Mass = 103.8 slugs

| Run | Initial Velocity<br>in/sec | Compensation<br>Coefficient | Exit Velocity<br>in/sec 4/20/98 | Exit Velocity<br>in/sec 6/27/95 |
|-----|----------------------------|-----------------------------|---------------------------------|---------------------------------|
| 1   | -0.250                     | .075                        | 0.253                           | 0.247                           |
| 2   | -1.000                     | .066                        | 0.995                           | 0.990                           |
| 3   | -3.000                     | .065                        | 2.911                           | 2.876                           |

Table 2 : Translational Compensation Results - Mass = 524.9 slugs

| Run | Initial Velocity<br>in/sec | Compensation<br>Coefficient | Exit Velocity<br>in/sec 4/20/98 | Exit Velocity<br>in/sec 6/27/95 |
|-----|----------------------------|-----------------------------|---------------------------------|---------------------------------|
| 1   | -0.250                     | .080                        | 0.250                           | 0.244                           |
| 2   | -1.000                     | .062                        | 1.008                           | 1.000                           |
| 3   | -3.000                     | .061                        | 2.970                           | 2.946                           |

Table 3 : Translational Compensation Results - Mass = 1070.6 slugs

Originally, rotational compensation tests were to be repeated using 2BODY and the leaf spring test article. However, previous rotational compensation tests using the leaf springs were not applicable to the CBM since a 21 Hertz torsion mode was present and excitable during tests with the CBM. Rotational compensation tests were carried out using a single pair of guides on the CBM ring as the contacting surfaces and a 21 Hertz digital notch filter. Using a single rotational degree of freedom configuration of 2BODY and a 2 millisecond cycle time, three different inertias were tested with various initial angular velocities. As seen in the past and in the range of test results exhibited in Tables 4 and 5, the lower inertias require more compensation, especially at the slower speeds. The results are also less repeatable at these lower velocities which often complicates the test process. Note that the current version of ROCKET requires less compensation than the previous version. However, given the range of results, it was decided to use the previous conservative compensation curves for rotation with the SGI based ROCKET.

| Run Date | Initial Velocity<br>deg/sec | Compensation<br>Coefficient | Minimum Exit<br>Vel. deg/sec | Maximum Exit<br>Vel. deg/sec |
|----------|-----------------------------|-----------------------------|------------------------------|------------------------------|
| 7/19/95  | -0.250                      | .125                        | 0.242                        | 0.408                        |
| 4/22/98  | -0.250                      | .105                        | 0.191                        | 0.251                        |
| 7/19/95  | -0.500                      | .115                        | 0.463                        | 0.534                        |
| 4/22/98  | -0.500                      | .081                        | 0.446                        | 0.490                        |
| 7/19/95  | -0.750                      | .110                        | 0.723                        | 0.760                        |
| 4/22/98  | -0.750                      | .080                        | 0.700                        | 0.753                        |
| 7/19/95  | -1.000                      | .107                        | 0.932                        | 1.002                        |
| 4/22/98  | -1.000                      | .077                        | 0.934                        | 0.936                        |

Table 4 : Rotational Compensation Results - Inertia = 405.4 sl-ft<sup>2</sup>

| Run Date | Inertia sl-ft <sup>2</sup> | Initial Velocity<br>deg/sec | Compensation<br>Coefficient | Exit Velocity<br>deg/sec |
|----------|----------------------------|-----------------------------|-----------------------------|--------------------------|
| 7/19/95  | 25300                      | -0.250                      | .10                         | 0.240                    |
| 4/22/98  | 25300                      | -0.250                      | .04                         | 0.242                    |
| 7/19/95  | 25300                      | -0.500                      | .086                        | 0.500                    |
| 4/22/98  | 25300                      | -0.500                      | .04                         | 0.426                    |
| 7/19/95  | 36200                      | -0.250                      | .11                         | 0.246                    |
| 4/22/98  | 36200                      | -0.250                      | .04                         | 0.268                    |

Table 5 : Rotational Compensation Results

The initial tests of ROCKET were performed in the contact type 2 mode of operation using the coil spring as the test article. The arm was initialized with a translational tip velocity perpendicular to the face of the CBM ring. The RMS joint motors maintained this velocity and direction until contact was first measured. At this point the arm was placed in test or limp mode and the peak impact force recorded. These test results are shown in Table 6 with previous runs using the same compensation curves. Although the results are limited, there is good correlation between the peak loads. Note that the later runs used a 2 millisecond cycle time while the earlier tests used 4 milliseconds.

| Payload             | Initial Velocity ft/s | Peak Force lb<br>3/25/97 | Peak Force lb<br>4/20/98 |
|---------------------|-----------------------|--------------------------|--------------------------|
| 1 (mass factor = 2) | -.03                  | 68.                      | 60.                      |
| 1 (mass factor = 2) | -.12                  | 275.                     | 257.                     |

Table 6 : ROCKET Coil Spring Results

The final set of ROCKET tests consisted of 2 ISS berths, payloads 9 and 17, using the CBM hardware. For each run, the brakes were off and payload mass factor was two. Again, the current version of ROCKET used a 2 millisecond cycle time while the earlier results were based on 4 milliseconds. The CBM was configured with only three latches and no thermal standoffs.

The payload 9 case had previously been run using the Alliant version of ROCKET with four latches and a complete set of thermal standoffs. However, in this run, three latches did most of the work during the berth and plunger contact occurred near the end of the run. Figure 1 is the berthing port separation vector for this run in

D2 coordinates. Figures 2 and 3 are the contact forces and moments measured during the run. Figures 4 through 6 are the berthing separation vector components in D1 coordinates for the SGI version of this run. Figures 7 through 12 are the corresponding force/moment sensor signals. Given the hardware and software differences between the two simulations, the results are still similar. The backdrive forces and moments computed for each run, though not shown here, are also in approximate agreement. Remember, the purpose of these runs was to detect obvious problems with the new math model. For completeness sake, the payload 17 results are shown in Appendix 1.

### **Conclusions / Recommendations**

The hardware / software integration activities between the real time docking and berthing simulation software and the hydraulic motion system and control panel of the MSFC 6DOF facility have been successfully completed. The results of these tests are a subset of the activities performed to verify the operation of the facility driven by the new SGI Challenge system host. The performance of the simulations has been enhanced not only by the reduced cycle time, but also through the elimination of the long dt which occurred in ROCKET at relinearization time. The results indicate that previous compensation curves are conservative with the new platform and may even be lowered. The new system appears to be functioning as well or better than the previous Alliant based simulation.



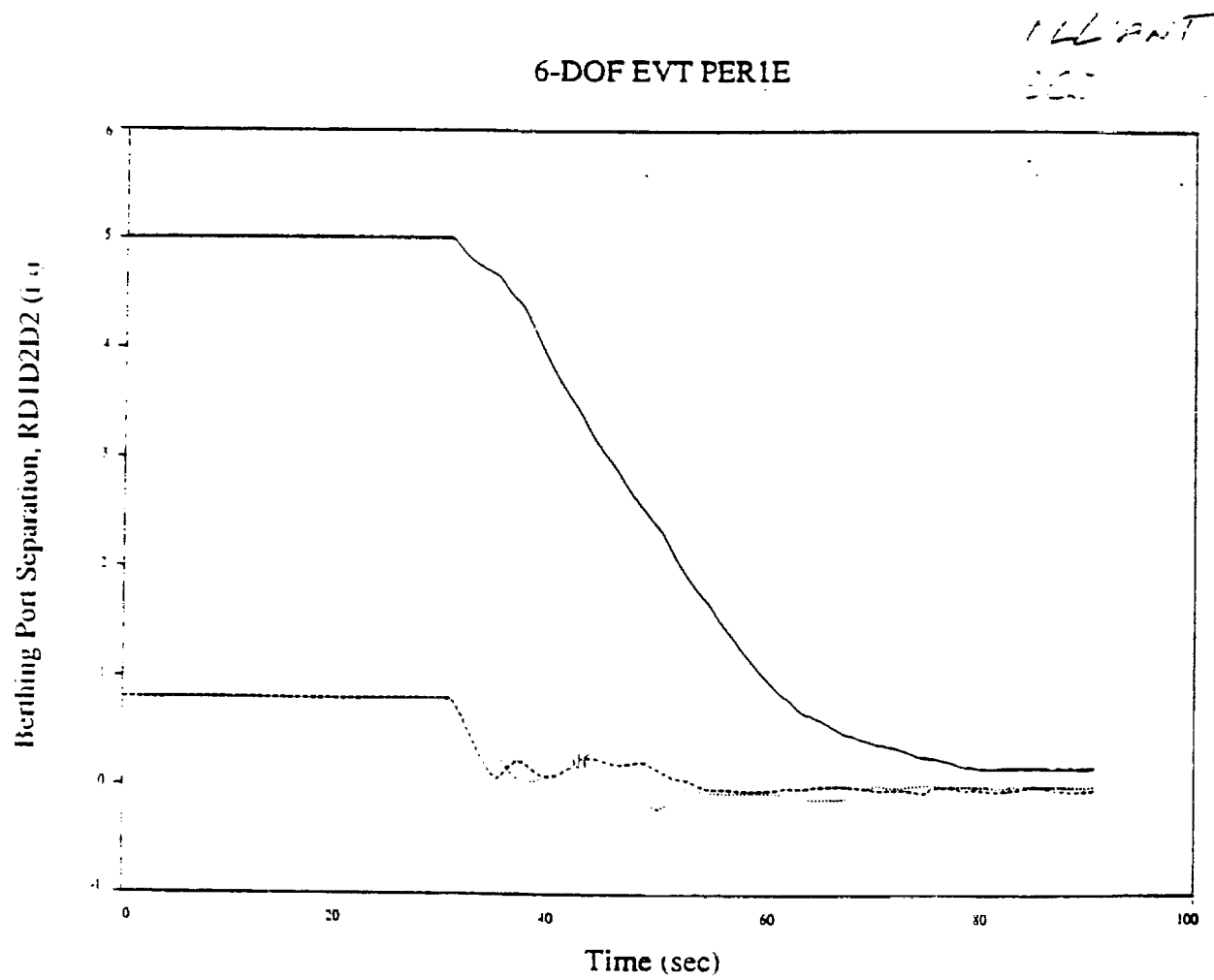


Figure 1

CR3L-A

Figure 2

# 6-DOF EVT PERIE - FSISI (1b)

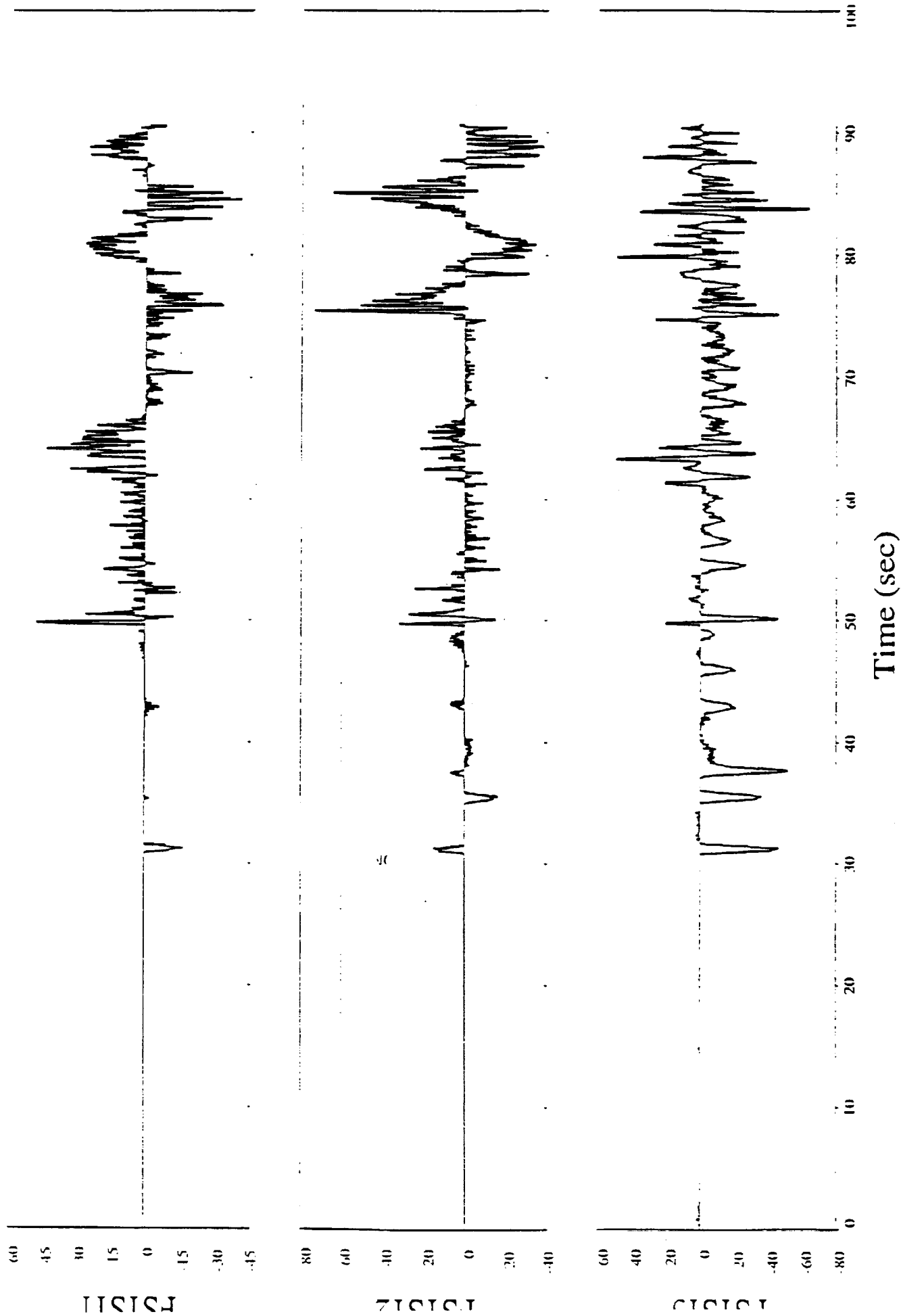
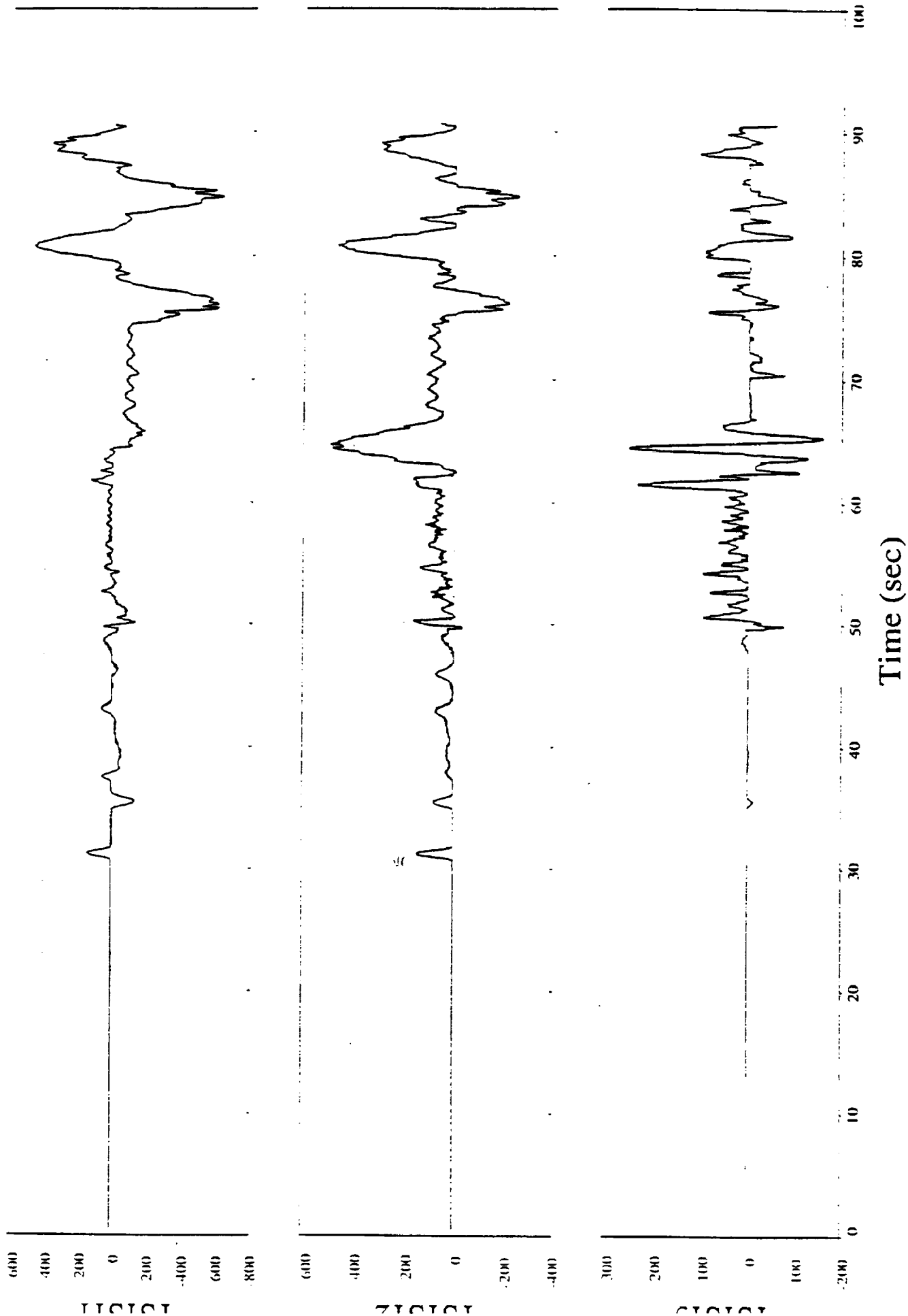


Figure 3

6-DOF EVT PERIE - TS1S1 (ft-lb)



CR3L\_A (pyld 9)

ct3l\_adl  
1 0 2

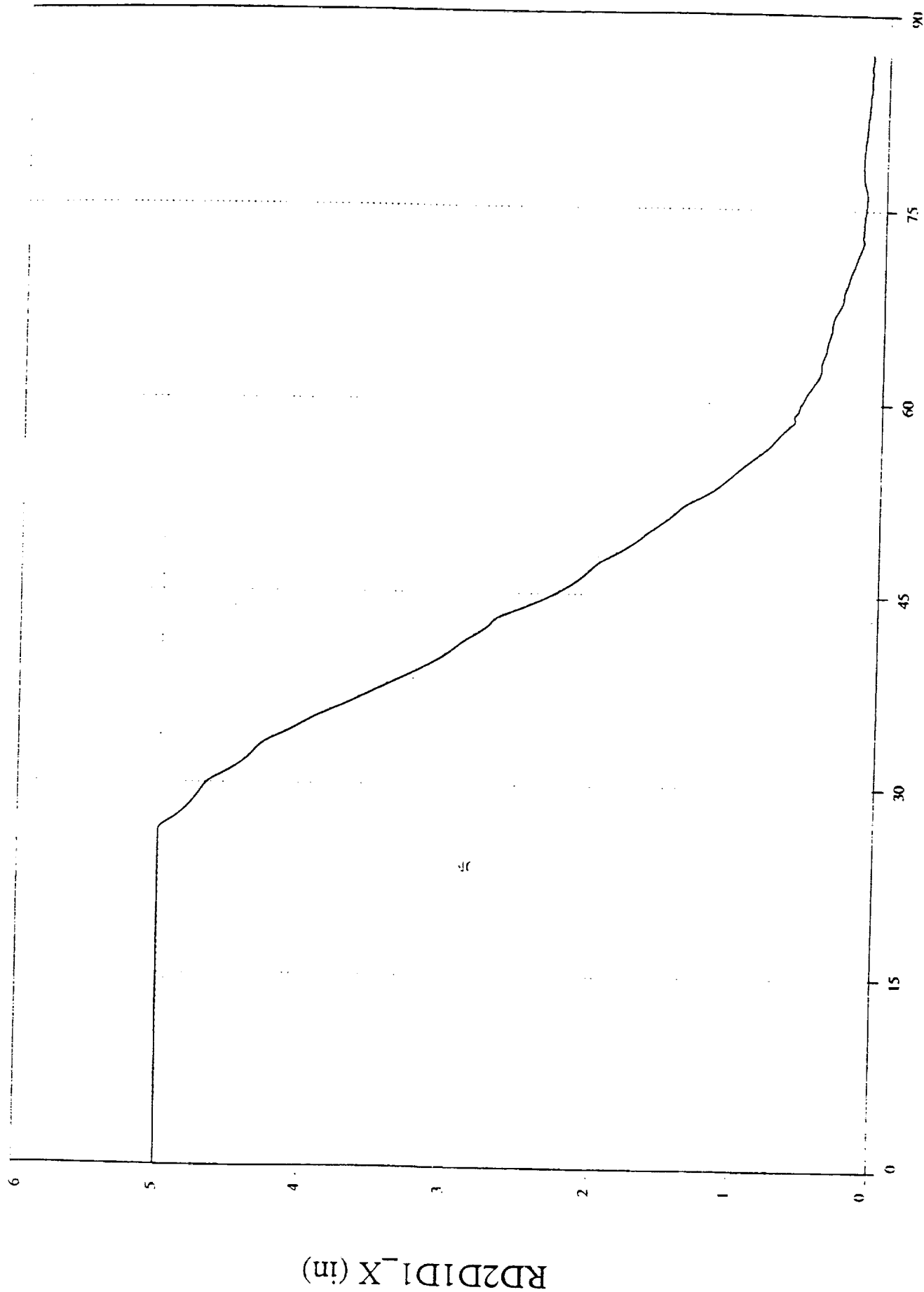


Figure 4

cr3l\_adl  
1 0 3

CR3L\_A (pyld 9)

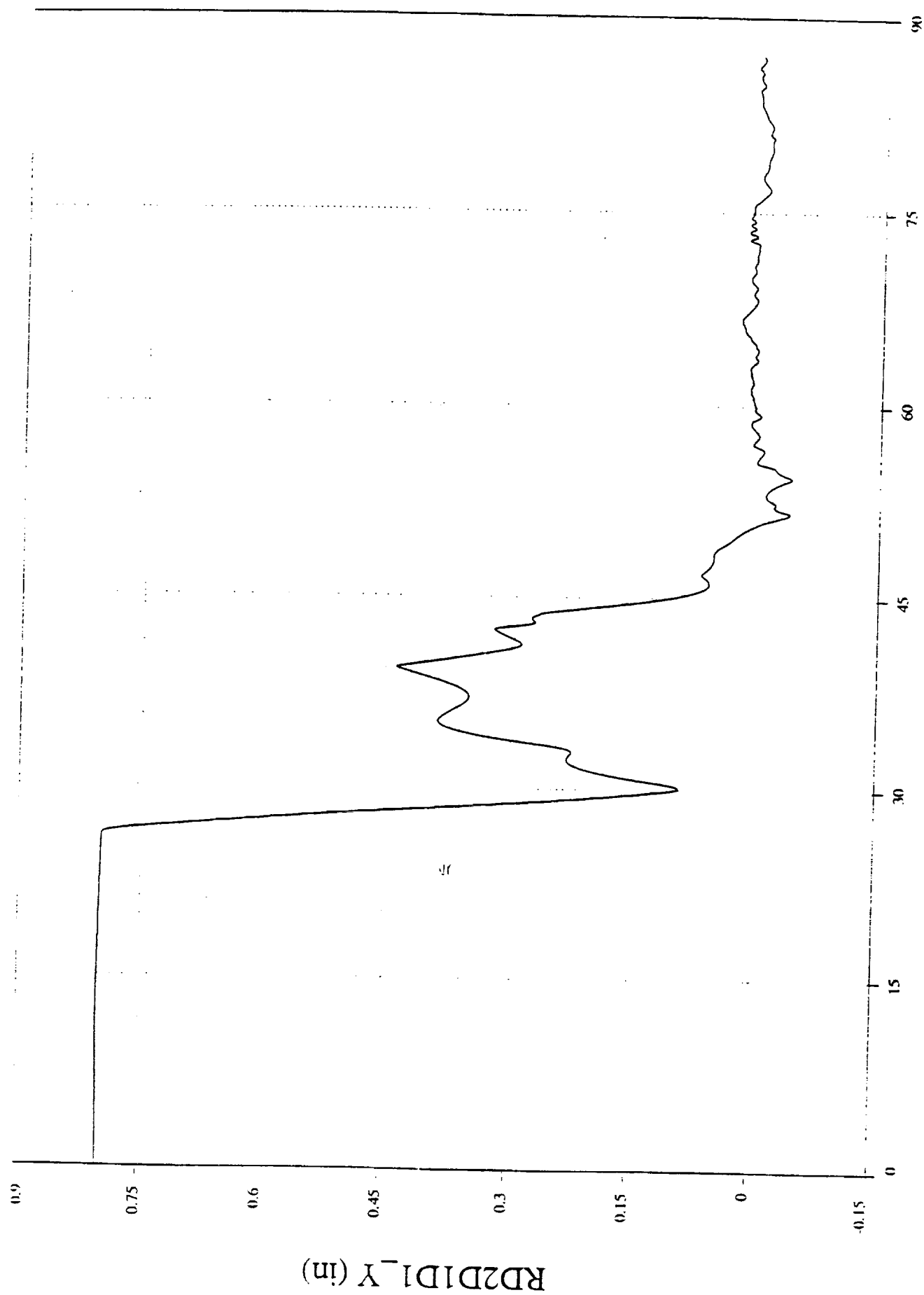


Figure 5

CR3L\_A (pyld 9)

cr3l\_adl  
1 0 4

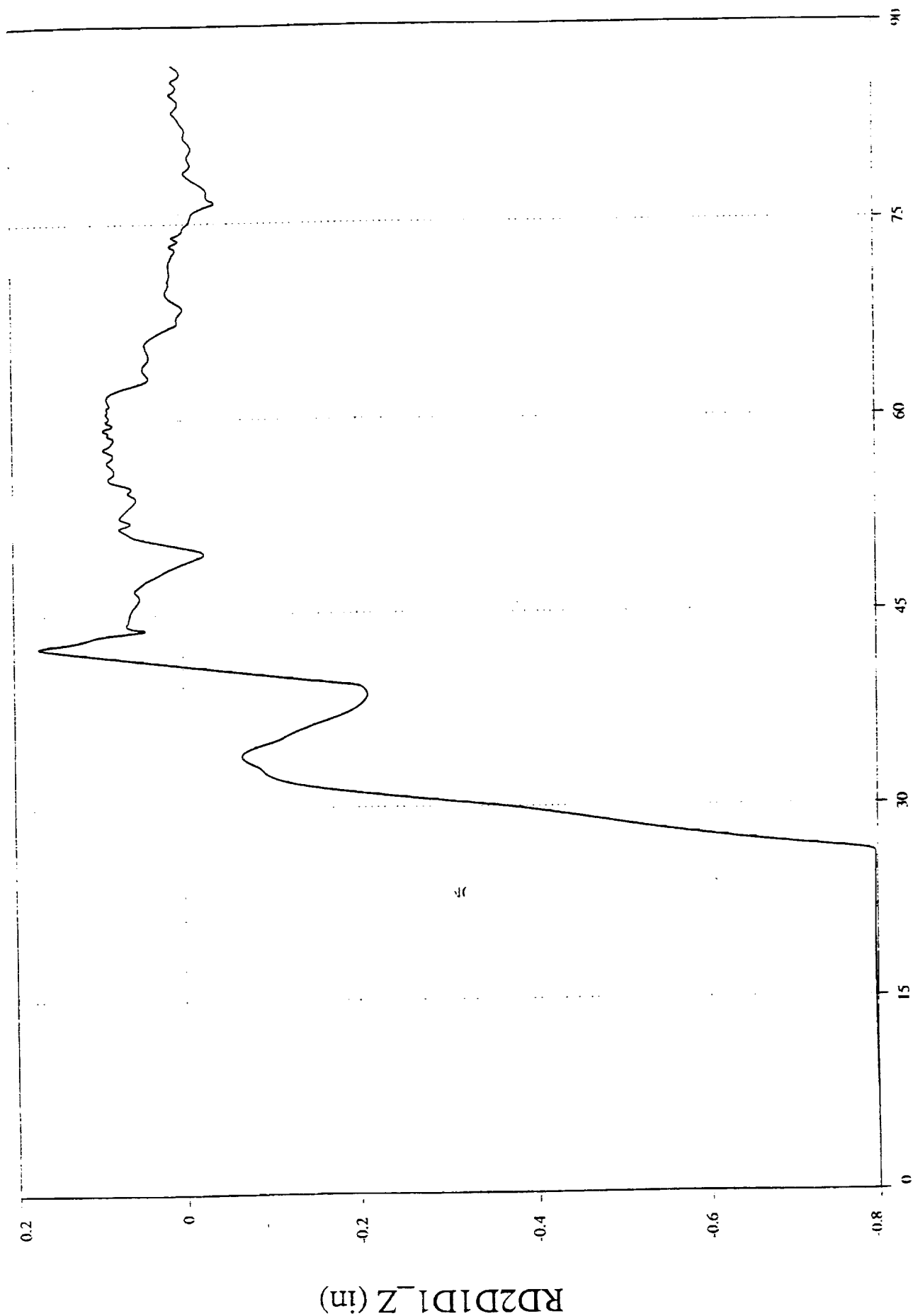


Figure 6

CR3L\_A (pyld 9)

cr3l\_ad1  
1 0 18

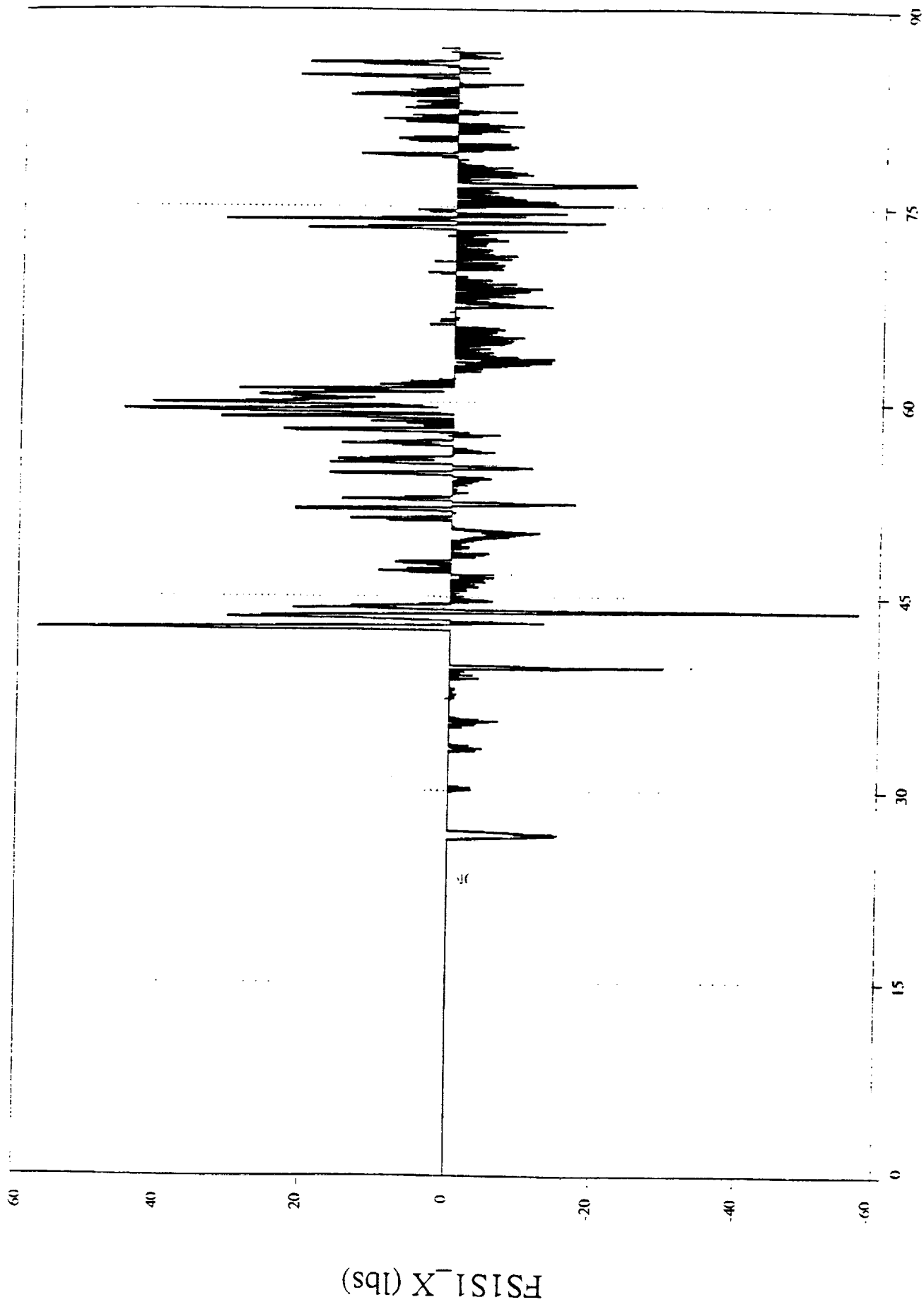


Figure 7

CR3L\_A (pyld 9)

cr3l\_adl 1 19

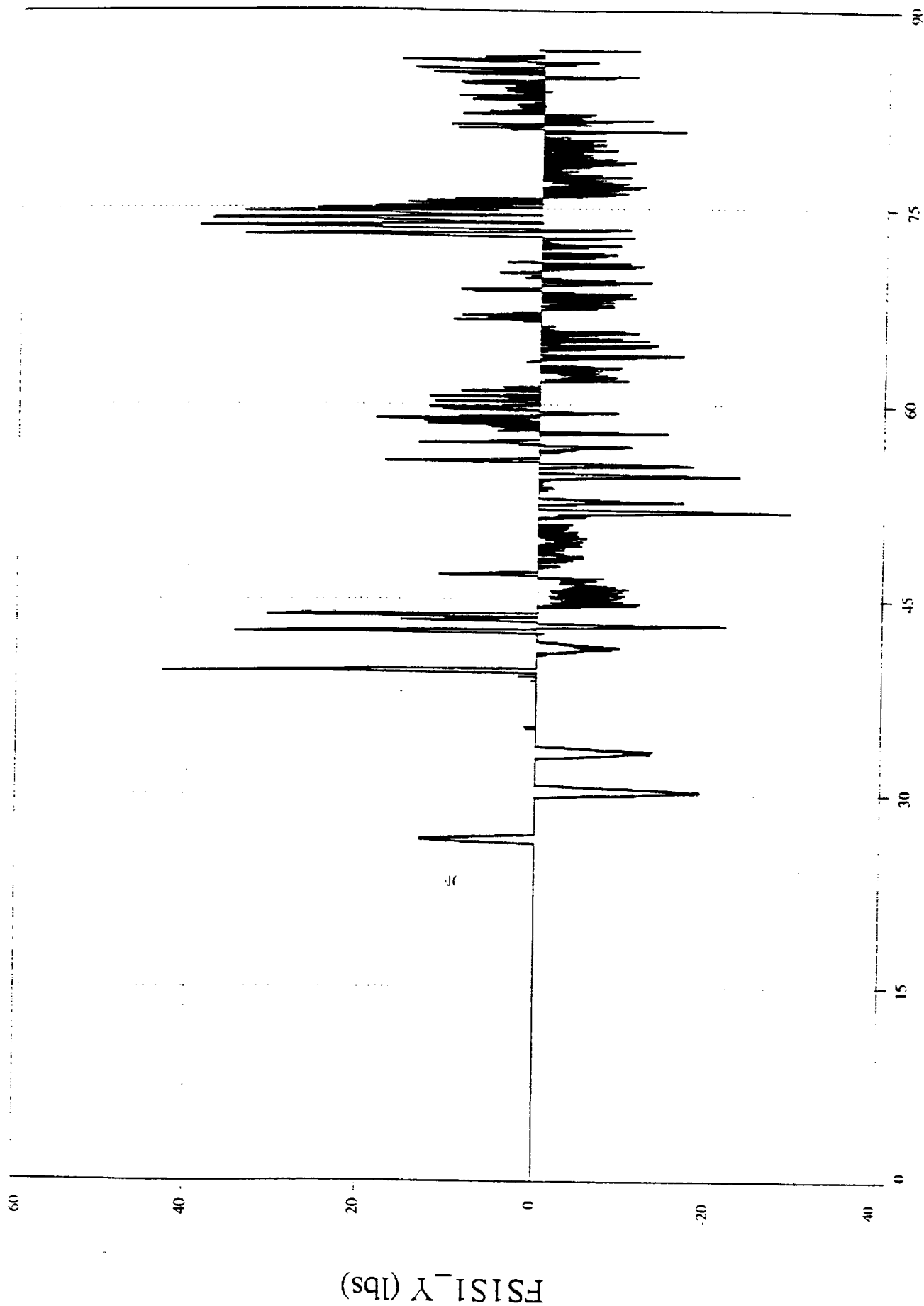


Figure 8



CR3L\_A (pyld 9)

cr3l\_adl

1 0 20

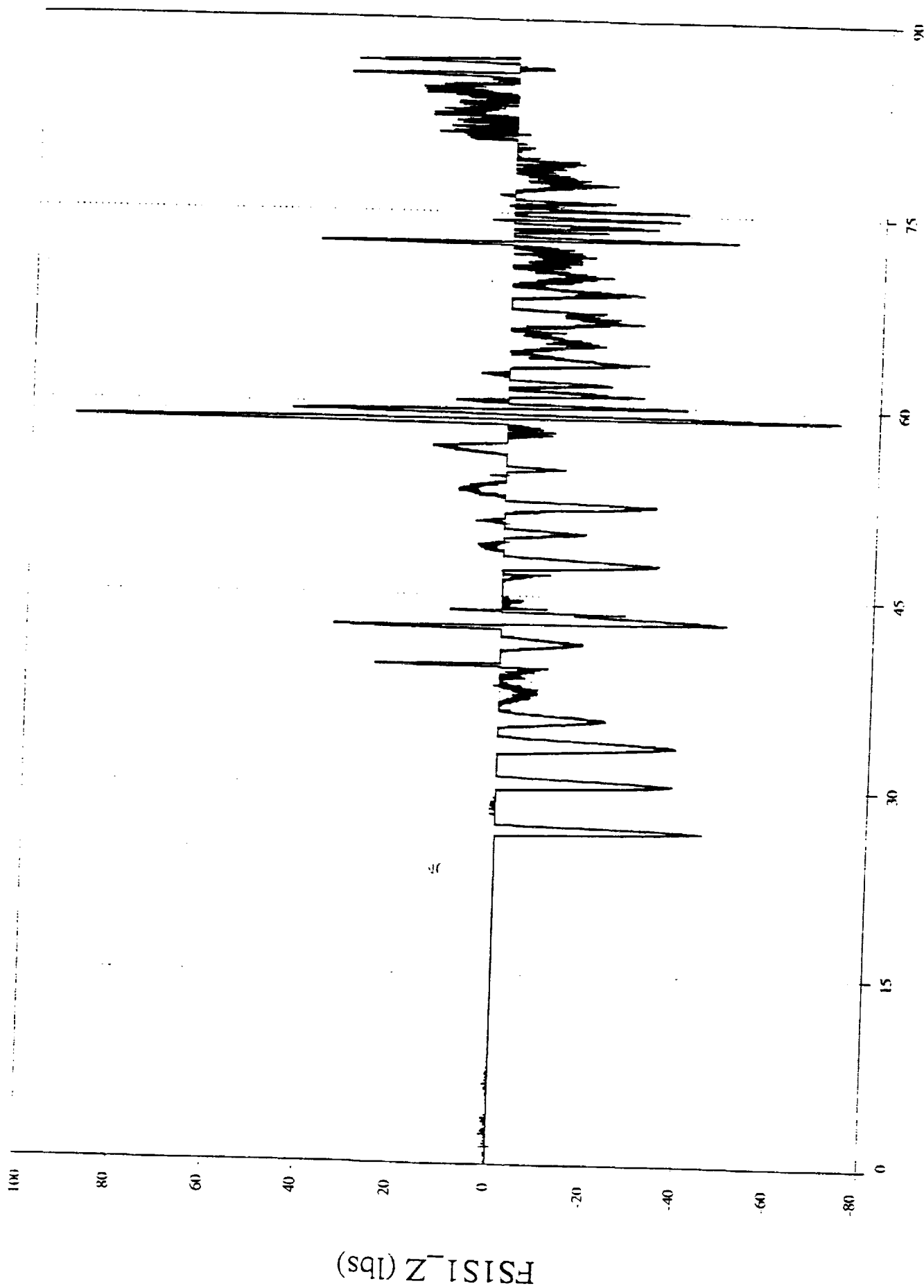


Figure 9

TIME(sec)

cr3l\_ad1  
1 0 — 21

CR3L\_A (pyld 9)

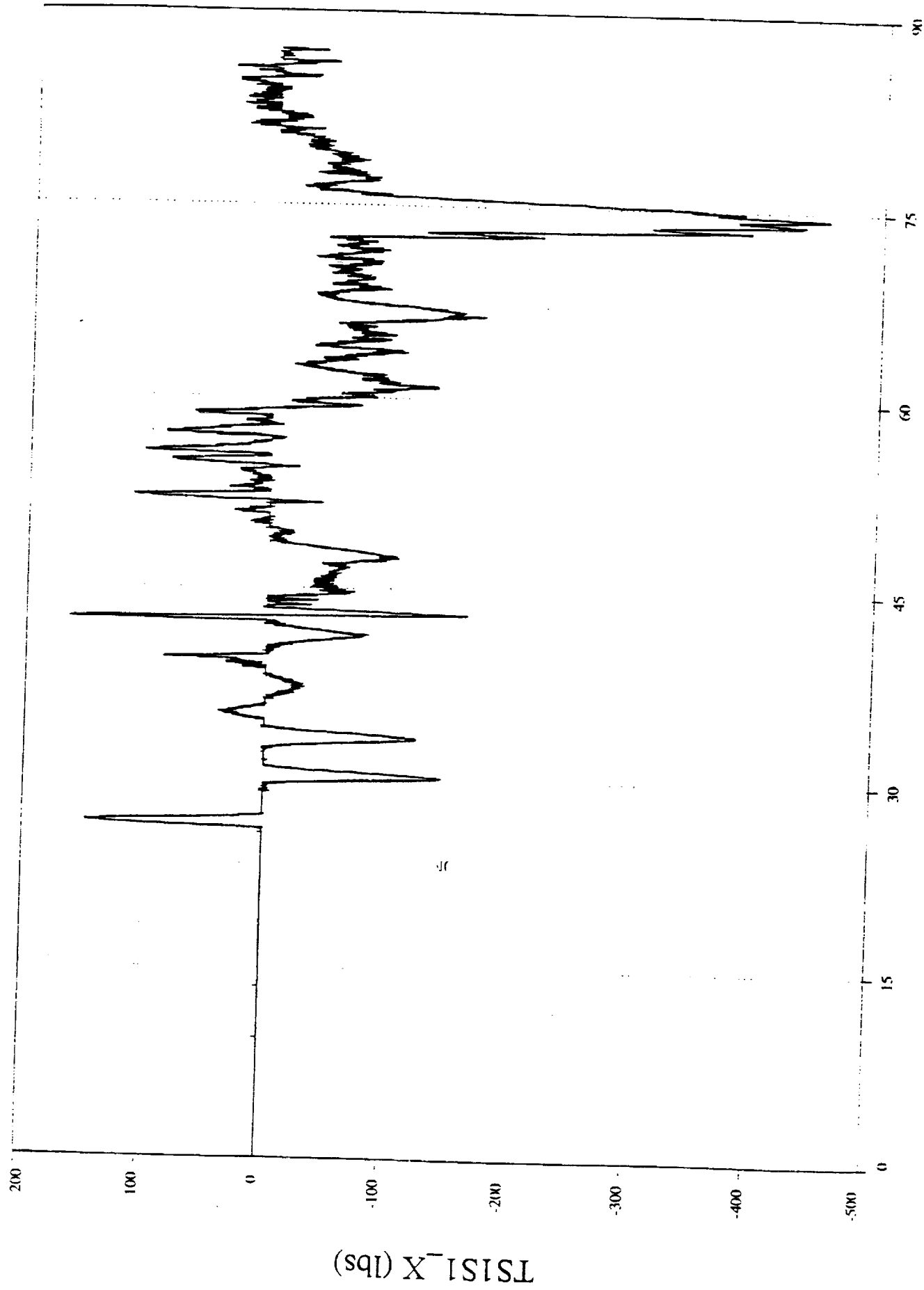


Figure 10

CR3L\_A (pyld 9)

cr3l\_adl 1 22

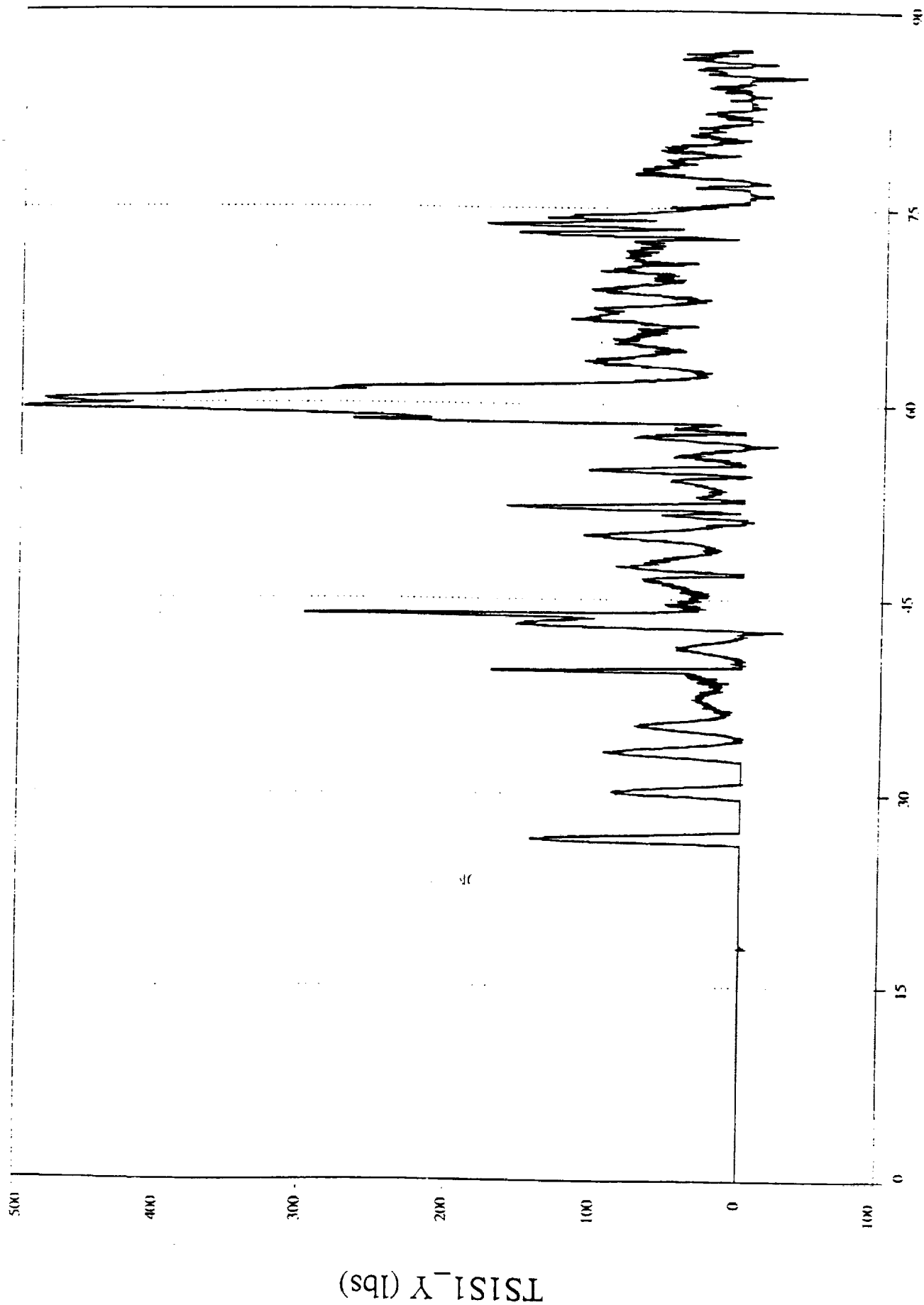


Figure 11

cr3l\_adl  
1 0 — 23

CR3L\_A (pyld 9)

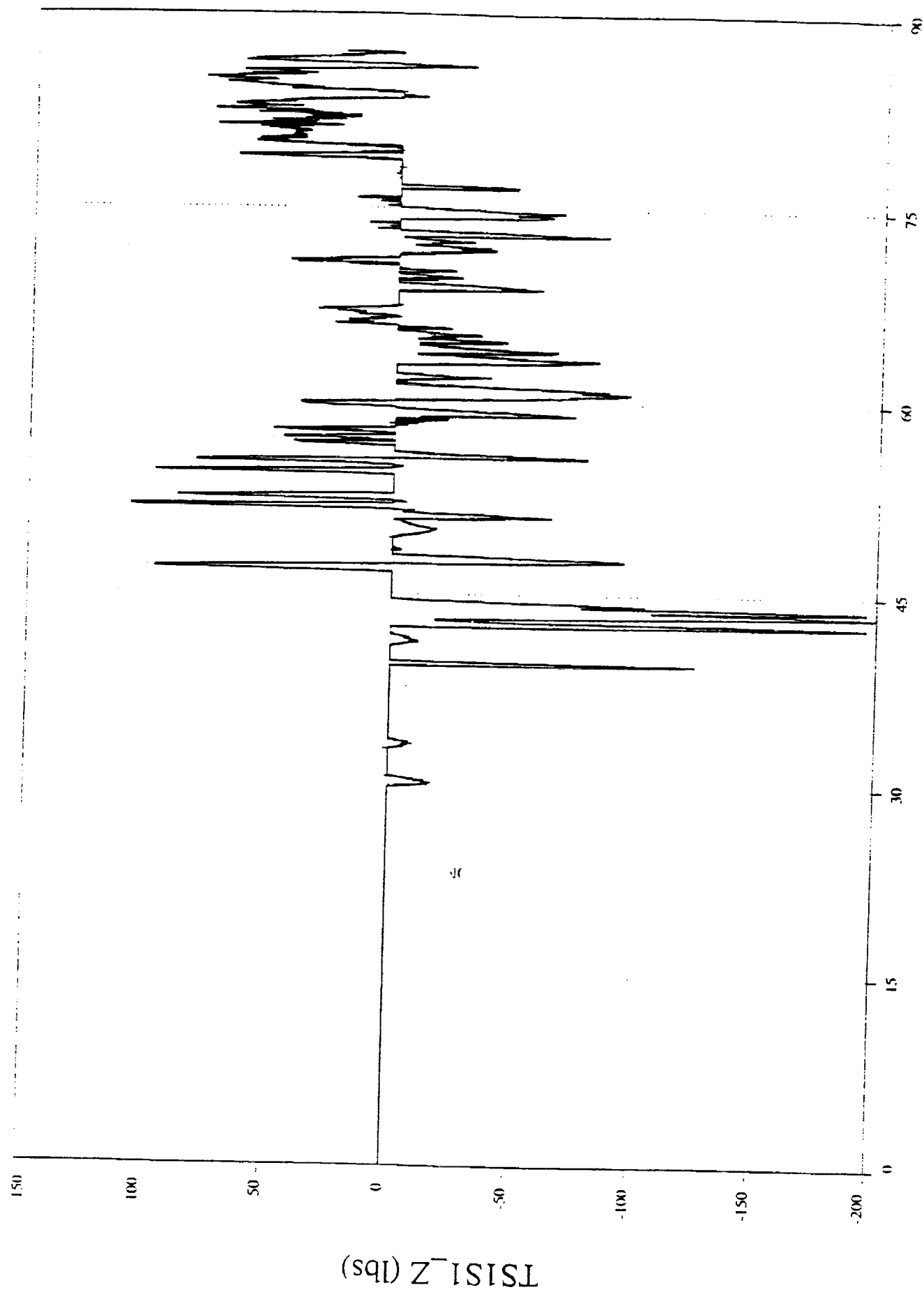
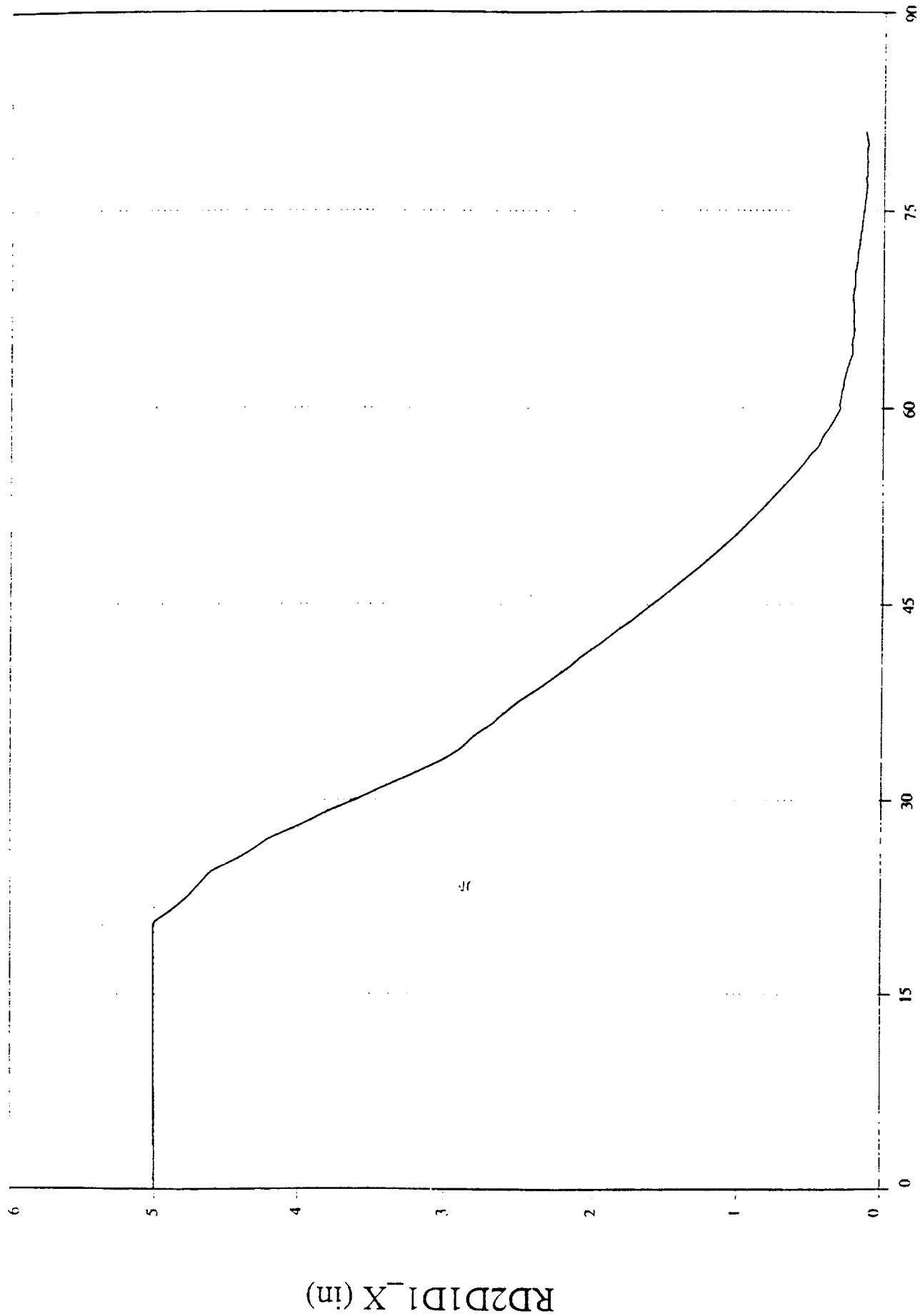


Figure 12

## Appendix 1

CR3L\_C (pyld 17)

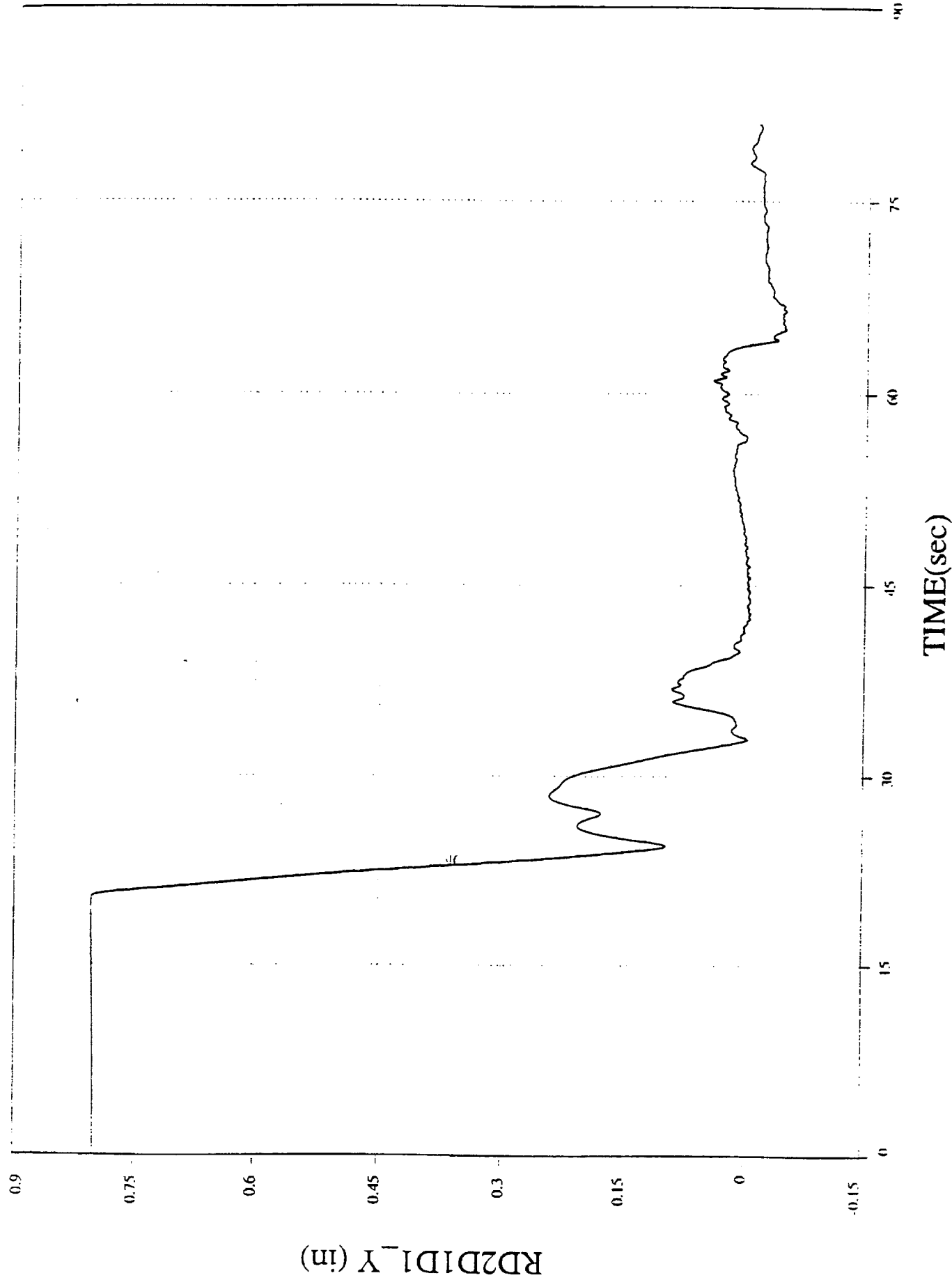
cr3l\_cdl  
1 0 2



TIME(sec)

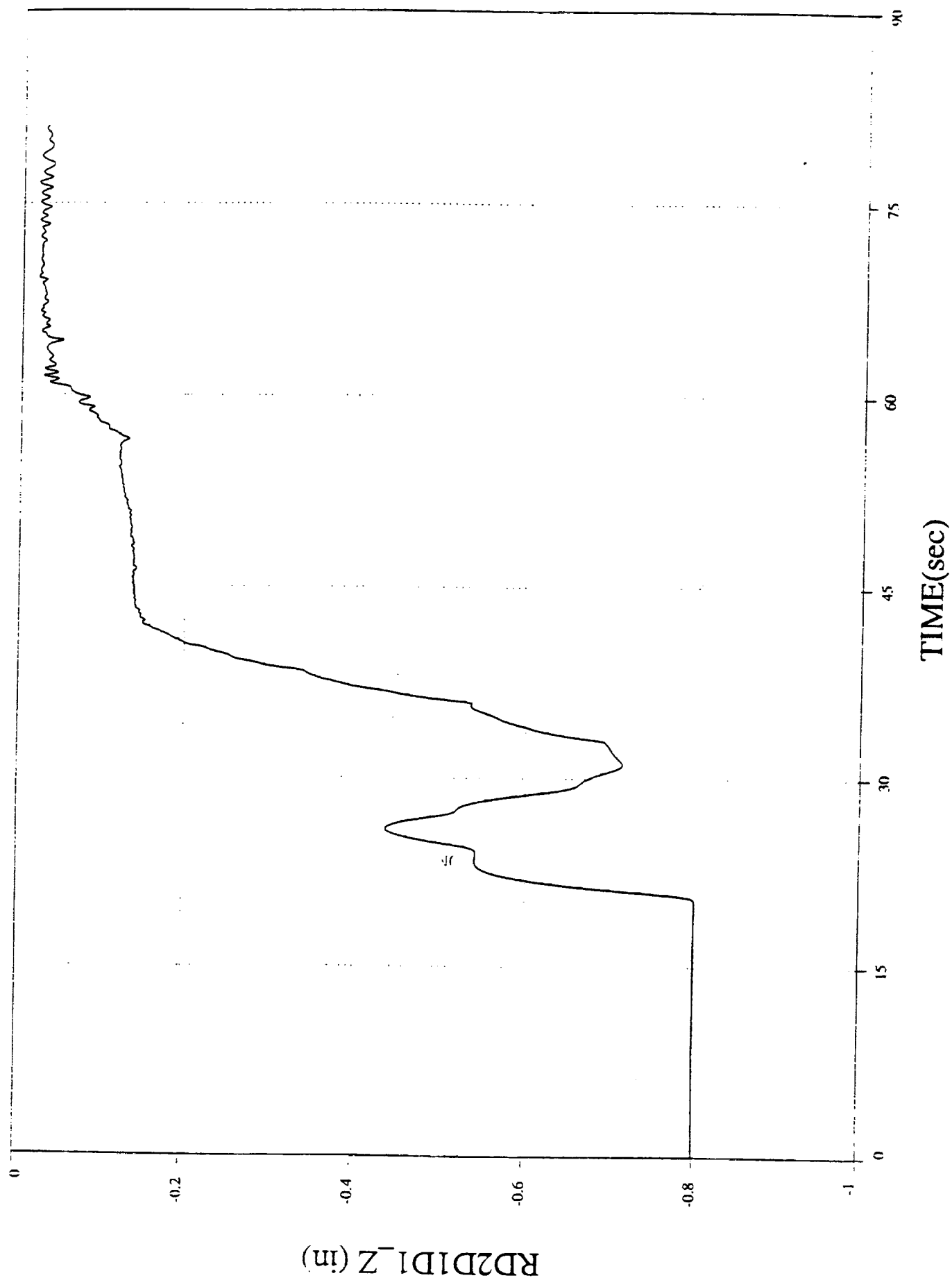
CR3L\_C (pyld 17)

cr3l\_cdl  
1 0 3



CR3L\_C (pyld 17)

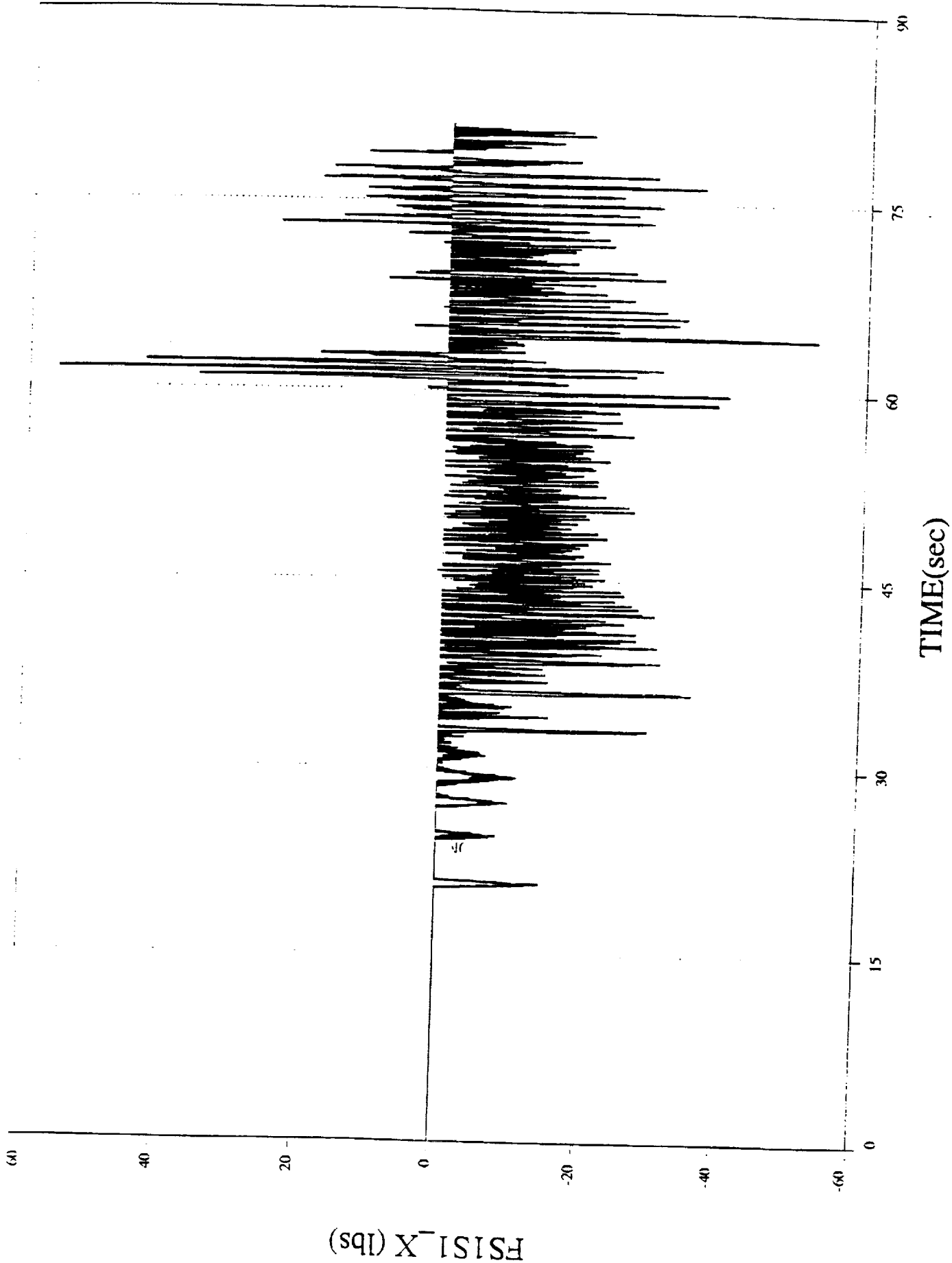
cr3l\_cdl  
1 0 — 4





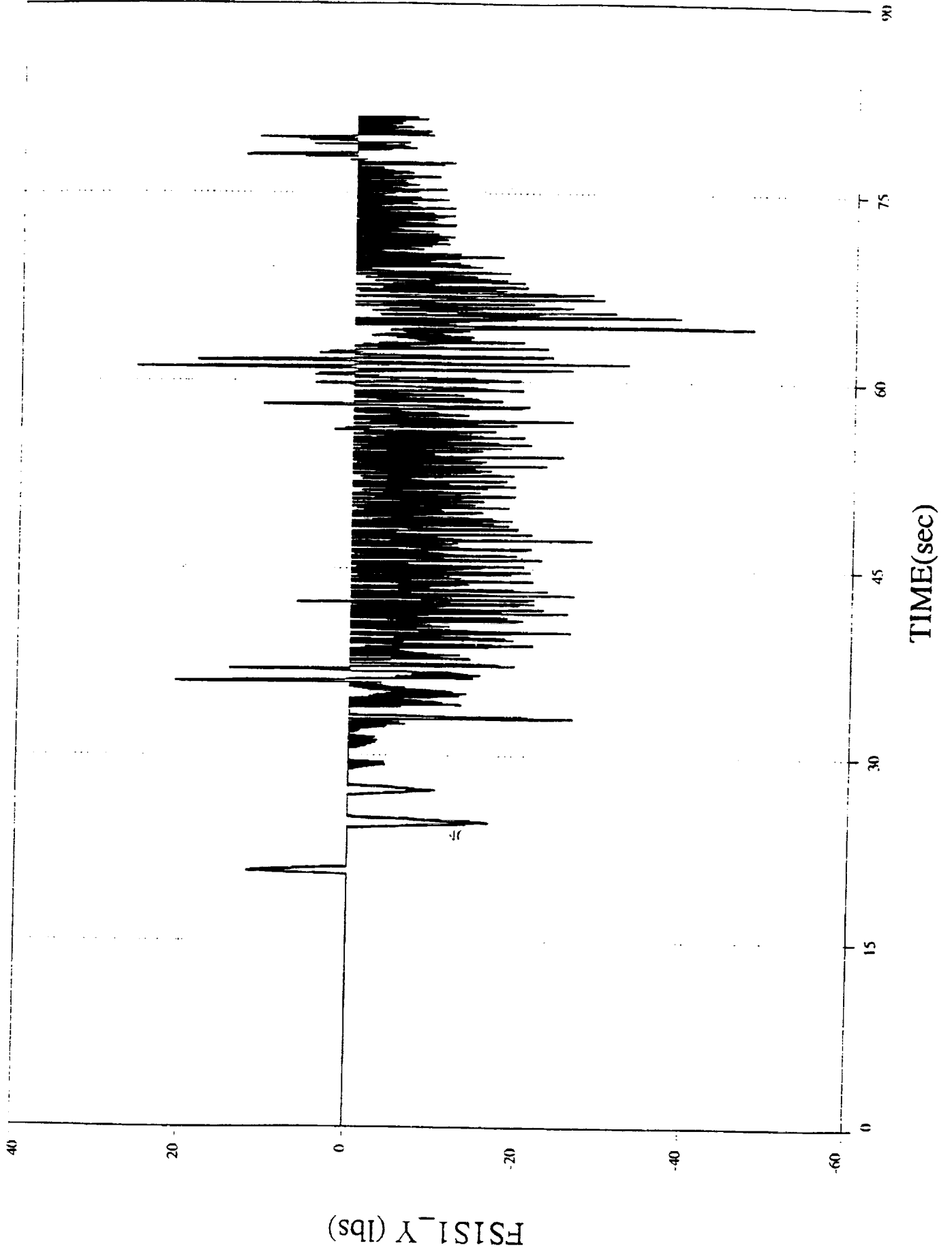
CR3L\_C (pyld 17)

cr3l\_cdl 18



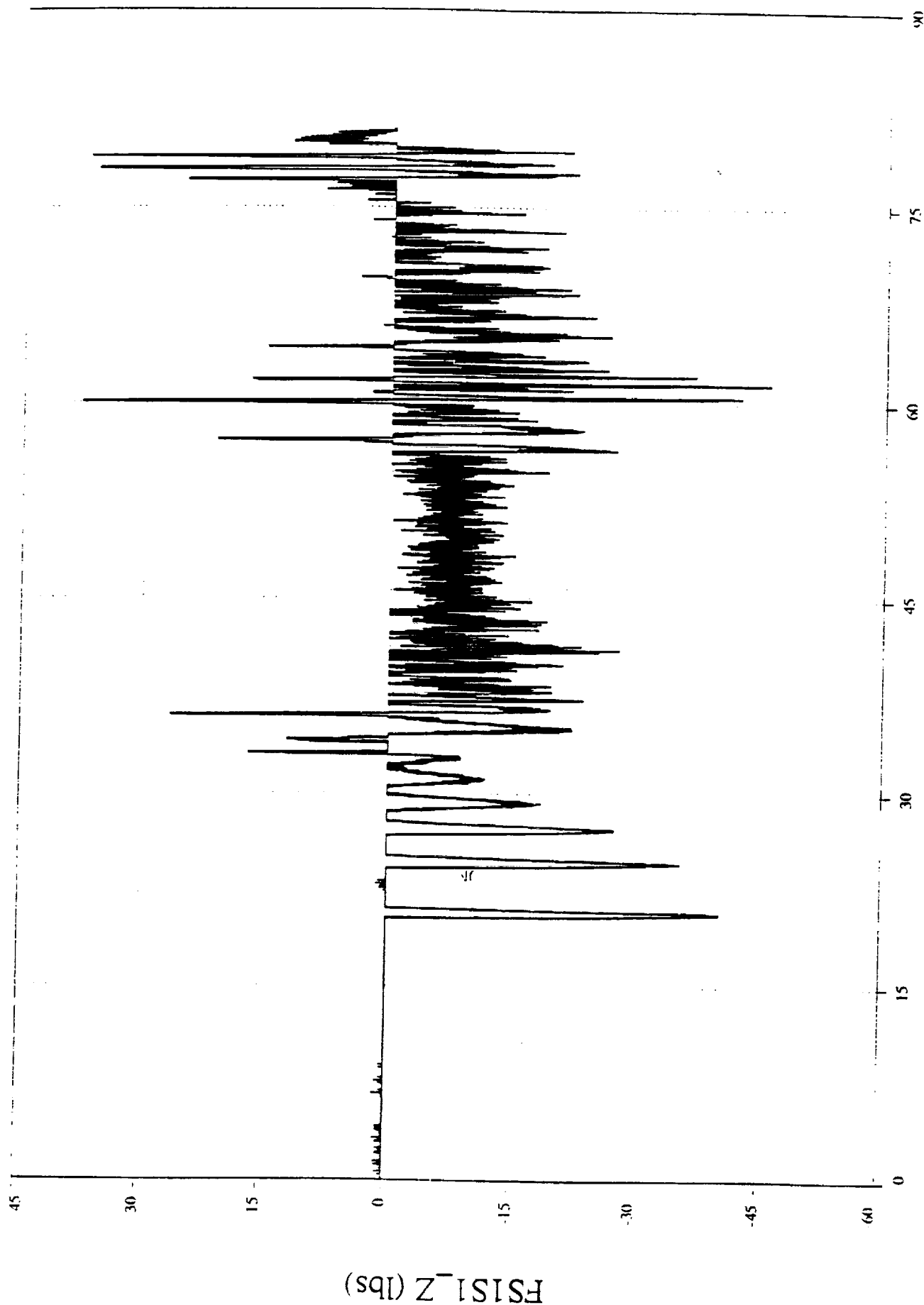
CR3L\_C (pyld 17)

Unit 19



CR3L\_C (pyld 17)

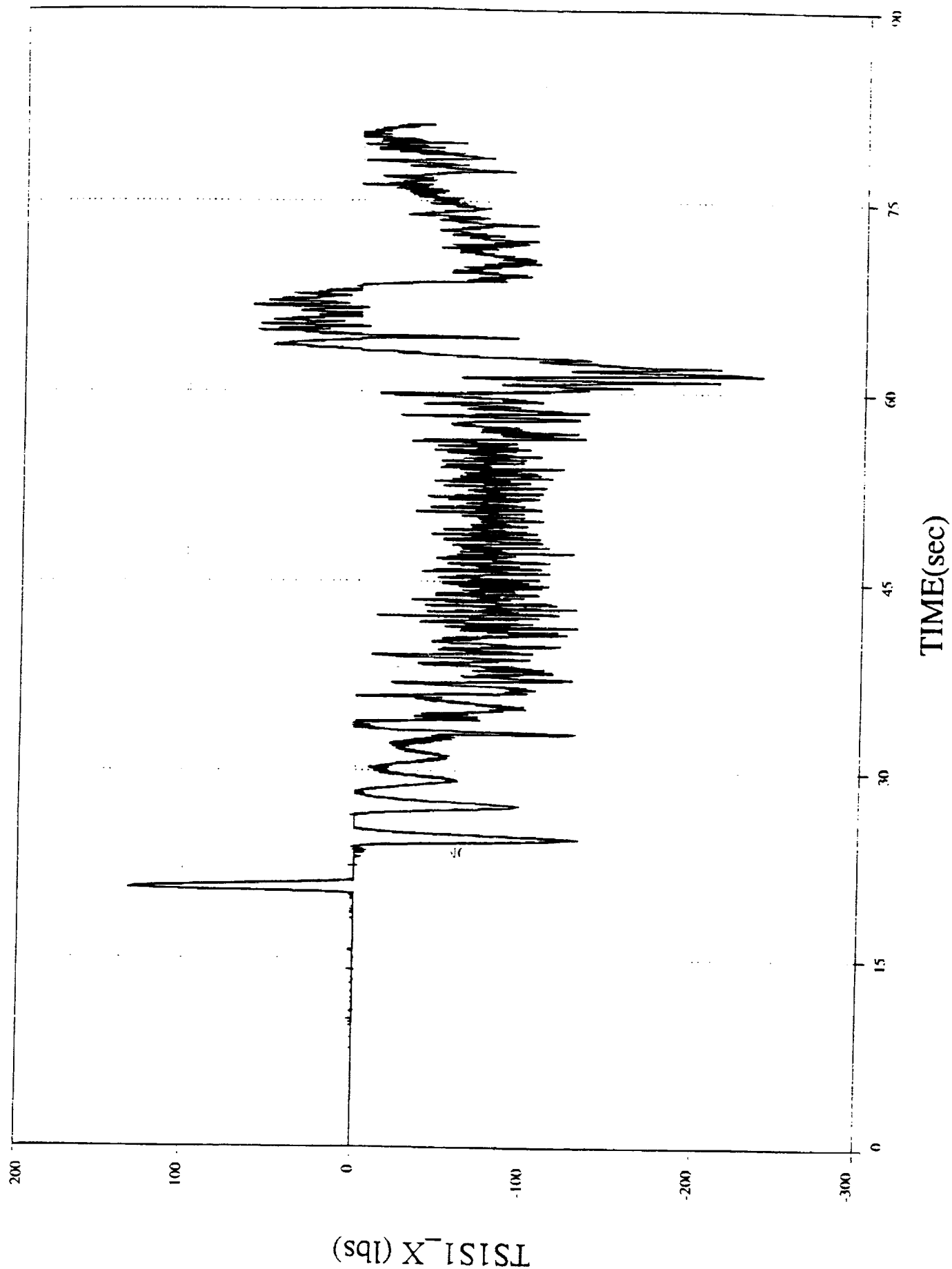
cr3l\_cd1  
1 0 — 20



TIME(sec)

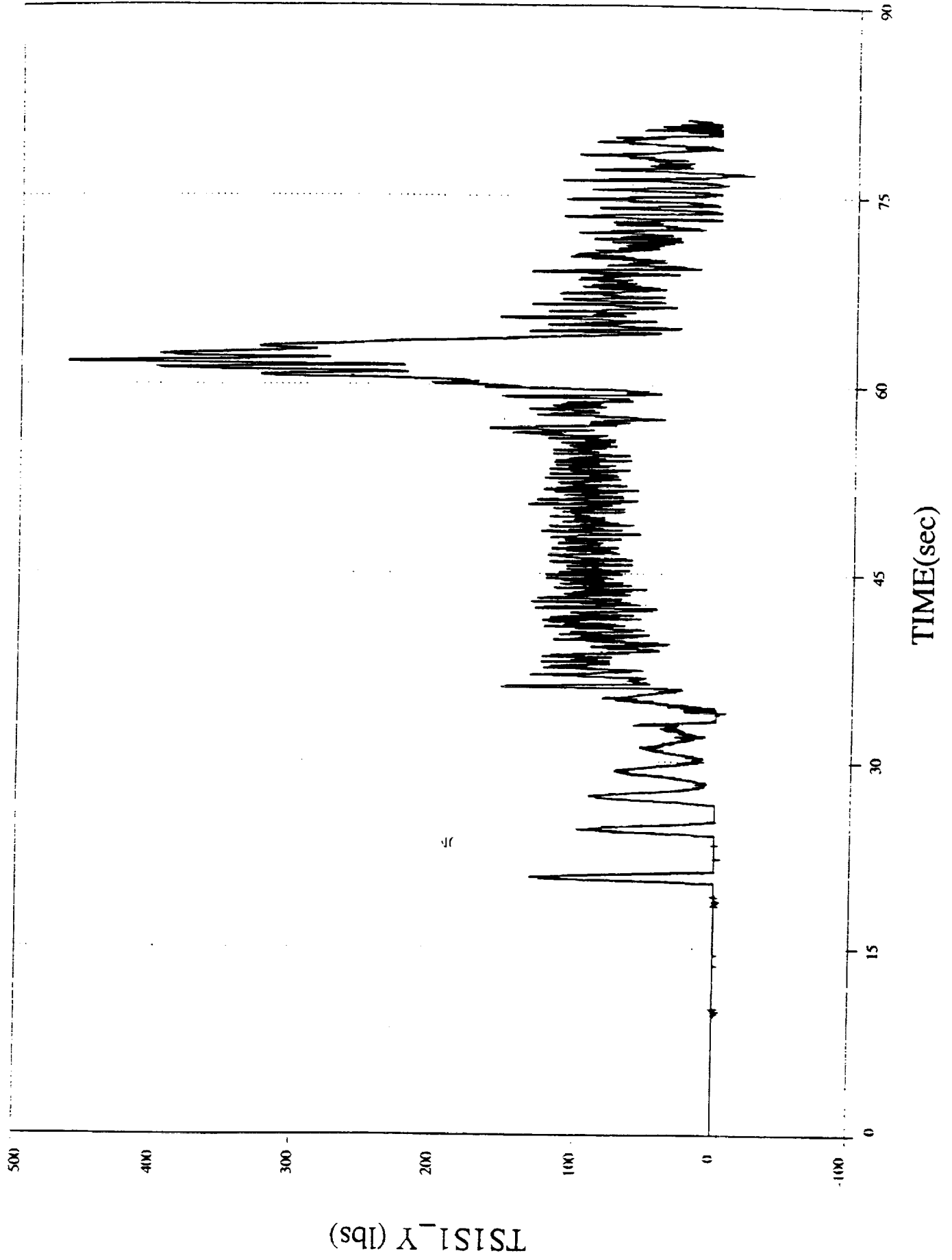
CR3L\_C (pyld 17)

cr3l\_cdl  
1 0 — 21



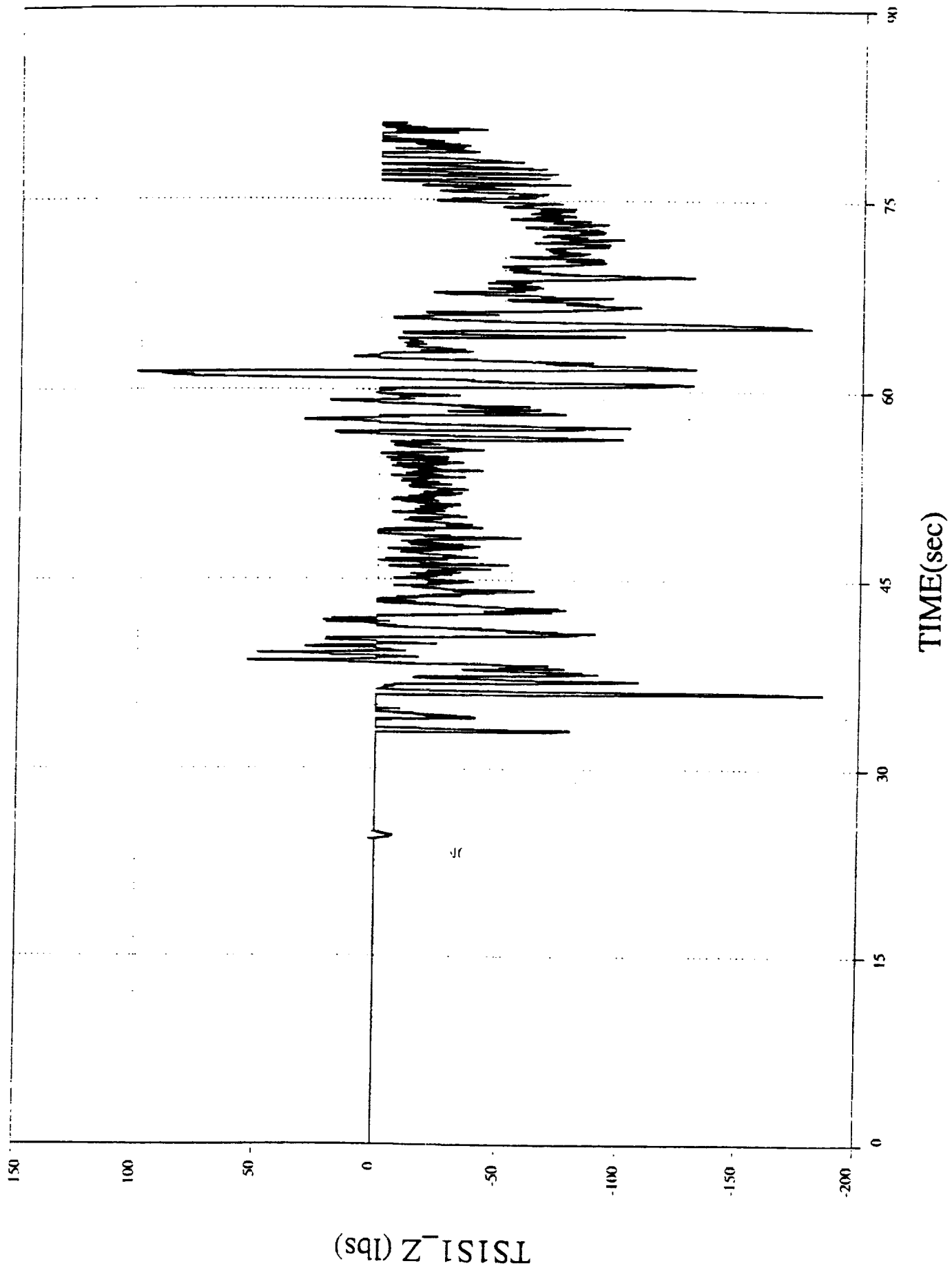
CR3L\_C (pyld 17)

cr3l\_cdl  
1 0 22



CR3L\_C (pyld 17)

cr3l\_cdl  
1 0 23



## Reference 12

DCI TM#082698-1  
Digital Servo Power Amplifier Model  
August 26, 1998

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts - Dr. Patrick Tobbe  
Subject : Digital Servo Power Amplifier Model

## Introduction

Recently the analog rate servo systems for the Remote Manipulator System (RMS) were replaced by digital controllers. This memorandum describes the digital Servo Power Amplifier (SPA) model development and integration into the real time RMS simulation *ROCKET* at the MSFC 6DOF facility. Block diagrams and algorithms for the SPA model were based on information from the SPAR document SPAR-RMS-SG.1936. These activities were necessary to meet the requirements of integrated CBM test 275.

## ROCKET Simulation Modifications

The RMS control system is shown in Figure 1. The GPC is the General Purpose Computer which supports the RMS flight software. The MCIU is the Manipulator Controller Interface Unit acting between the GPC and the Arm Based Electronics (ABE). The previous version of *ROCKET* did not simulate the MCIU and corresponding time delays between the components of the controller. MCIU and digital servo models provided by SPAR were coded by bd Systems and Dynamic Concepts and integrated into *ROCKET*. New interfaces between the GPC flight software and joint servo models were developed to accommodate the MCIU and maintain the original analog servo module. The MCIU passes rate commands from the GPC to the ABE, and encoder and tachometer signals from the ABE back to the GPC. The MCIU and encoder modules operate at a 42 millisecond cycle time. The GPC functions at an 80 millisecond rate.

The modifications to *ROCKET* consisted of the development of five new subroutines and changes to several others. The five new routines are *MCIU*, *ENCODER*, *MOTOR*, *ANALOG\_SPA*, and *DIGITAL\_SPA*. Modifications were made to *RMS*, *SERVO*, *CONTACT*, *INTFAC*, *S\_DYN.INC*, and *CTLINIT*.

The functions of subroutine *SERVO* have been distributed between *MCIU*, *MOTOR*, *ANALOG\_SPA*, and *DIGITAL\_SPA*. *SERVO* is no longer used in *ROCKET*. The executive routine *RMS* of the dynamics process was modified to incorporate the new routines. Two new timing variables, *DYN.T42* and *DYN.T42\_CYCLE*, were introduced to control the timing of *MCIU* and *ENCODER*. *ENCODER* and *MCIU* are called by *RMS* before the rate servo logic every 42 milliseconds based on the value of *DYN.T42*. At this time, the flag *DYN.SAVE\_TACH* is also toggled to true. This flag is used by the SPA routines to sample and hold the outer control loop tachometer signal *DYN.TACH\_OLOOP*. The SPA routines are then called as part of the 2 millisecond loop. However, these routines will be called four to eight times in a row in order to decrease the step size used in these algorithms and increase the number of samples for the tachometer and current signals. The number of sequential calls for these routines will be determined through timing tests. The minimum number of calls required for a .5 millisecond rate is four, while



the goal for a .25 millisecond sample rate is eight calls. The flag *DYN.ANALOG\_SERVO* set in *RMS\_INIT* determines whether *ANALOG\_SPA* or *DIGITAL\_SPA* is called. The new routine *MOTOR* is executed every 2 milliseconds after the series of SPA calls.

The subroutine *ENCODER* discretizes joint angles from the state vector *DYN.XCUM* and stores these values in the internal array *I\_ANGLE\_ENCODE*. This array is then used to populate the delayed integer array *DYN.I\_ENCODE\_DELAY* for use by *MCIU*. This function was deleted from *INTFAC*.

*MCIU* delays *DYN.I\_ENCODE\_DELAY* another 42 milliseconds and outputs the delayed values in the floating point array *DtoC.J\_ANGLE\_ENCODE* used by the GPC flight software. *MCIU* also delays the discretized outer loop tachometer array *DYN.TACH\_OLOOP* from the SPA routines. The delayed tachometer signal output to the GPC is *DtoC.TACH\_OP*. *MCIU* also delays and discretizes motor rate commands from the GPC for use by the SPA routines. The GPC flight software command *DtoC.J\_FXPT\_RATE\_CMD* is limited, scaled, and converted into the fixed point integer array *DYN.YL*. This conversion was previously done in *SERVO*.

The SPA routines use the joint rate commands *DYN.YL* from *MCIU* and motor currents *DYN.MOT\_CURRENT* from *MOTOR* to compute motor voltage commands *DYN.MOT\_VOLTAGE\_CMD*. Motor rates from *DYN.MOT\_SPEED* are the inputs to the tachometer models in the SPA routines.

*ANALOG\_SPA* was derived from the subroutine *SERVO*. It is the model of the original analog SPA control laws, shown in Figure 2, used on the RMS. In addition to the motor voltage commands, *ANALOG\_SPA* also computes the forward / backdrive flags *DYN.FFB*. These variables are used in *MOTOR* to determine motor torque limits.

*DIGITAL\_SPA* is the model of the digital SPA control laws developed from Figure 3. It uses the same inputs as *ANALOG\_SPA* to compute the motor voltage commands and forward / backdrive flags. It also requires the current attenuation command array, *Ia*, and control gain set identifier, *DYN.GAIN\_SET*. Both of these variables are payload dependant. *Ia* will become a global variable computed by the GPC with the introduction of POHS. However, it is currently set to its nominal value of zero in the initialization section of *DIGITAL\_SPA*.

The subroutine *MOTOR* was extracted from the original *SERVO* module and is based on Figure 4. This subroutine computes the motor torque applied to the gear box, *DYN.MOT\_TORQUE*, from the SPA input voltage and the motor shaft rate. The back EMF voltage is calculated from *DYN.MOT\_SPEED* and the gain *DYN.KB*. The motor current, *DYN.MOT\_CURRENT*, is developed from the motor torque and motor torque constant.

All new global variables required for the SPA upgrades were added to the structure *S\_DYN.INC*. The subroutines *CTLINIT* and *CONTACT* were modified to initialize the delayed sensor variables.

## **Conclusions / Recommendations**

The new digital SPA model for the RMS has been incorporated into *ROCKET*. The capability to perform runs using the analog SPA has been retained. A motor module has been extracted from the original *SERVO* file to provide a generic interface for the SPA models. Time delays between the GPC and ABE have been introduced using a simplified model of the MCIU. Further work is required to perform validation runs using the digital SPA. Upon integration of POHS into the flight software, the current attenuation command array *Ia* should be added to the *HCCF* module.

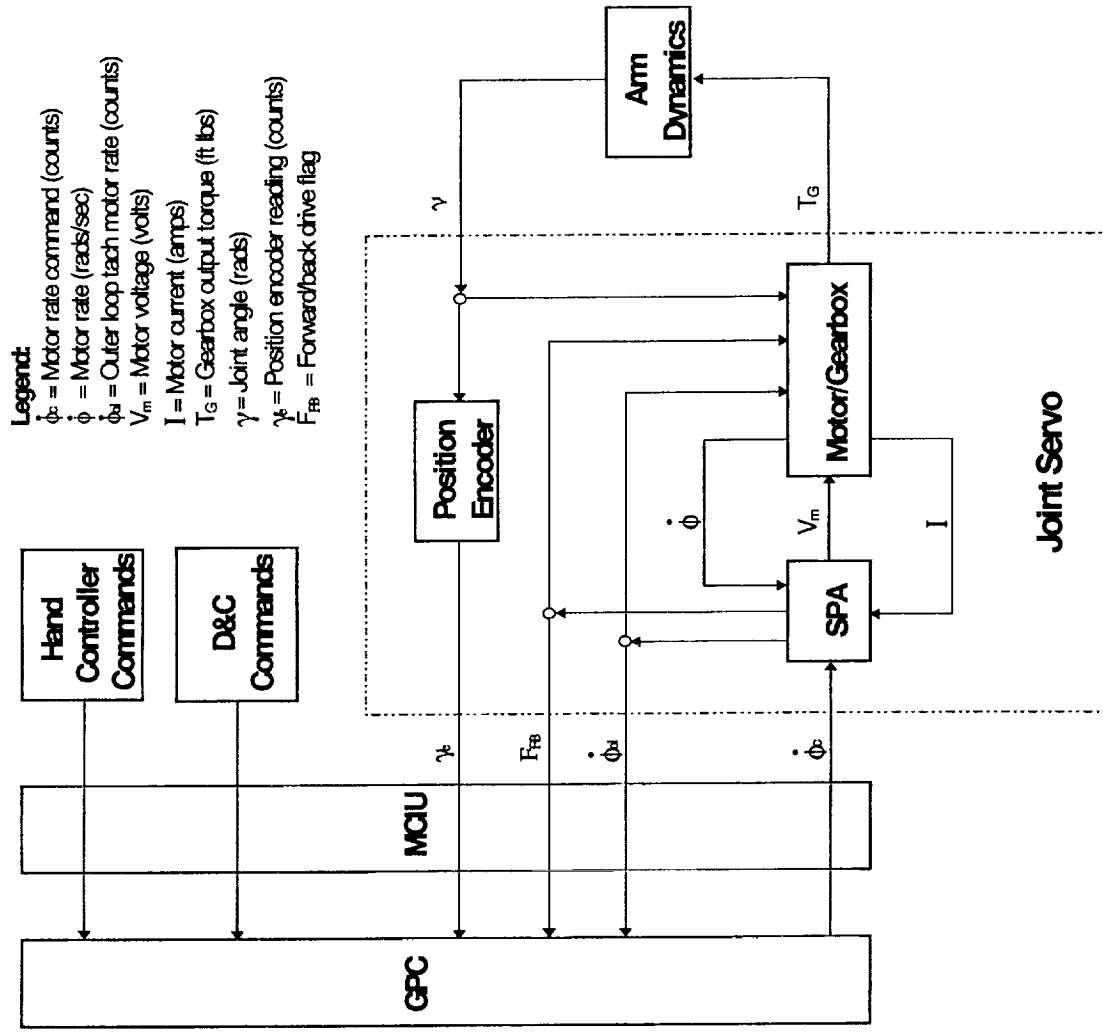


Figure 1 :RMS Control Model Block Diagram

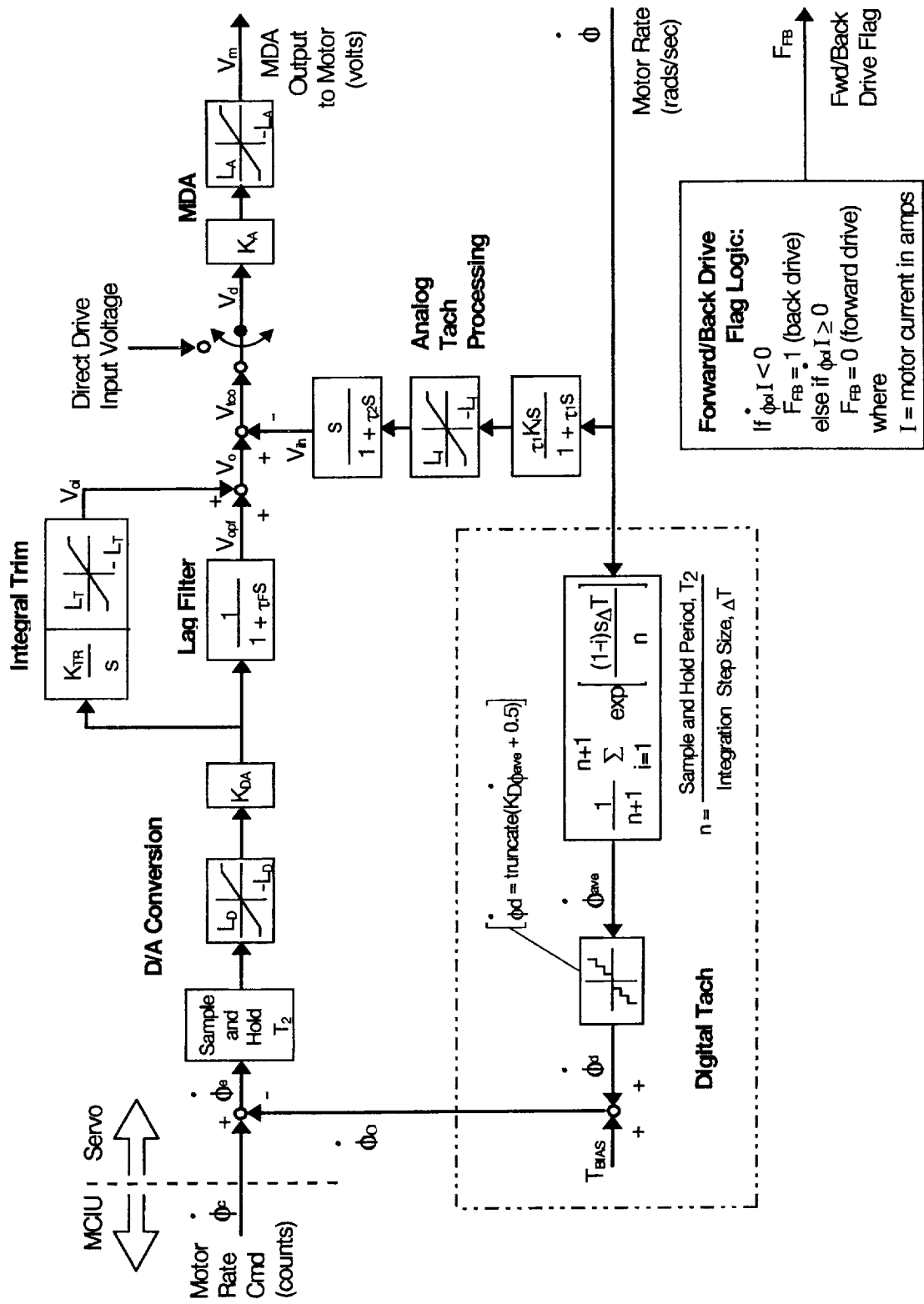


Figure 2 : Analog SPA Block Diagram





bd Systems®  
TCD20000103A  
30 April 2000

Contract No.  
NAS8-40604  
Final Technical Report

## Reference 13

DCI TM#103198-1  
RMS Simulation POHS Upgrade  
October 31, 1998

## Dynamic Concepts Technical Memorandum

#103198-1

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts - Dr. Patrick Tobbe  
Subject : RMS Simulation POHS Upgrade

### Introduction

This memorandum describes the changes made to the real time RMS simulation *ROCKET* in order to integrate Position Orientation Hold Select (POHS) capability into the flight software. As part of this activity, all modes of operation of *ROCKET* were re-organized to operate under flight software control. Several flight software and simulation initialization routines were modified or deleted to accomplish this goal. These activities were necessary to meet the requirements of integrated CBM test 275.

### ROCKET Simulation Changes

The *ROCKET* simulation was modified in three major areas of operation : active arm control, capture, validation. All three of these simulation modes now run under flight software control as documented in Functional Subsystem Software Requirements (FSSR) OI-26. Initialization and operating routines were substantially changed for all three of these simulation modes.

Active arm control changes primarily involved upgrades to the flight software routines in the control (CTL) process to incorporate the POHS capability. This was accomplished through a thorough comparison of the latest FSSR with the *ROCKET* simulation routines. Based on the differences found, it was decided to modify the existing routines using the block diagrams in the FSSR. The bulk of the changes were in the subroutines *EXEC*, *MIL\_INIT*, *CTF*, *HCAD*, *KDG*, and *DDP*. The previous routine *EXEC\_MIL* was re-named *EXEC* and modified to incorporate a simulated arm run away mode. The routines *EXEC\_CON* and *EXEC\_VAL* were deleted. *EXEC* fully supports RMS control mode transitions. The subroutine *MIL* was renamed *MIL\_INIT* to initialize man-in-the-loop operations. *MIL\_INIT* was changed to incorporate initial calls to the *ENCODER*, *MCIU* and *EXEC* routines. In addition to performing coordinate frame transformations of arm tip velocity commands, *CTF* now corrects for undesired arm motion. This is the principal function of POHS. Any arm displacements not along the desired trajectory result in additional tip velocity commands to counteract this motion. All control gains for POHS are input as I-loads in the *PAYLOAD* subroutine. *HCAD* was modified to move the fly-to / fly-from manual operations from the payload mode to the loaded orbiter mode. The fly-to / fly-from joy stick commands are now expressed in the orbiter based frames OBAS and ORAS and no longer in the CBM D1 frame. The subroutines *KDG* and *DDP* were upgraded to include data display variable calculations. These variables include sensed and commanded POR position, attitude, velocity, and angular velocity vectors. The outputs of *KDG* and *DDP* were then interfaced to *SGI\_SEND* to be displayed on the graphical RMS control panel.



In order to more accurately model the capture portion of the berthing process, the *ROCKET* contact operations were placed under flight software control. The previous contact mode of operation was re-named capture. There are five capture modes : idle, test, cartesian / tip position hold, joint position hold, dynamic. The RMS is in idle mode for brakes on, safing, software stop, and no mode entered conditions. For the capture process in idle, all brakes are engaged on the joints. All brake operations for the *ROCKET* dynamic process are now performed in *MCIU* based on commands from the flight software of the control process. This change also lends itself to new brake models in the future. The capture process under test mode is the previous brakes-off or limp mode. Here the arm is in test mode in the flight software and the motor current is attenuated to near zero. Therefore, there is some resistance from the joint motors in test mode in addition to inertia and friction. Cartesian / tip position hold mode makes use of the new POHS software. Here the arm is placed in the MITL orbiter loaded mode and POHS is enabled. For zero joystick commands, the arm will default into a position hold mode. Tip velocity commands are computed to maintain the current POR position and attitude. The joint position hold mode generates joint rate commands proportional to joint position error in order to maintain the current POR position and attitude. For this mode the arm is also placed under the MITL orbiter loaded mode, but POHS is disabled. The dynamic mode of operation is the same used under the previous contact mode. The arm is initialized to a specified relative velocity between the CBM berthing port frames and commanded to maintain this rate until contact is sensed. At this point, the flight software transitions to test mode. As part of these software changes, the subroutine *CONTACT* was replaced by *CAPT\_INIT*. *CAPT\_INIT* not only initializes the arm and motor state vectors, but also calls *ENCODER*, *MCIU*, and *EXEC* to compute initial flight software variables. Also, as mentioned earlier, the subroutine *EXEC\_CON* was deleted and all flight software commands for the capture process are computed in *EXEC*.

In order to streamline *ROCKET* and operate under a single flight software executive routine, the validation code was dramatically altered. An event scheduler was written to ease the validation process for current and future test cases. Two new variable arrays were declared, *VAL\_EVENT\_TIME* and *VAL\_EVENT\_ACTION*, to drive the flight software for validation cases. *VAL\_EVENT\_TIME* is an array of times which signal a change or event in the controlling process. *VAL\_EVENT\_ACTION* is an array of RMS control actions to be carried out at times designated by *VAL\_EVENT\_TIME*. Table 1 is the list of RMS actions and corresponding values used by *VAL\_EVENT\_ACTION*. All validation initialization software was removed from *INTERACT* and placed in the new subroutine *VAL\_INIT*. For each validation case, *VAL\_EVENT\_TIME* and *VAL\_EVENT\_ACTION* are set in *VAL\_INIT*. The previous event times, *VT\_CSTART*, *VT\_CEND*, *VT\_BRAKES*, and *VT\_TERM* have been deleted. In the validation mode of operation, the new subroutine *VAL\_SCHED* is called before *EXEC* to direct flight software actions. *VAL\_SCHED* simply uses simulation time, *VAL\_EVENT\_TIME*, and *VAL\_EVENT\_ACTION* to set specific flight software variables at the appropriate times.

| VAL_EVENT_ACTION | Flight Software Action                  |
|------------------|---|
| 1                | Idle Mode                               |
| 2                | Test Mode                               |
| 3                | Single Joint Mode                       |
| 4                | Direct Drive Mode                       |
| 5                | MITL - Orbiter Unloaded                 |
| 6                | MITL - Orbiter Loaded                   |
| 7                | MITL - Payload                          |
| 8                | MITL - End Effector                     |
| + / - 9          | Brakes On / Brake Off                   |
| 10               | Safing On                               |
| 11               | Single Joint / Direct Drive Command Off |
| 12               | Terminate Run                           |
| + / -13          | Coarse / Vernier Scaling                |
| + / - 14         | Command Direction + / -                 |

Table 1 : Validation Event Scheduler

## Conclusions / Recommendations

The POHS RMS flight software capability has been integrated into *ROCKET*. As part of this process, the modes of operation of *ROCKET* were re-organized to function under flight software control. The *ROCKET* simulation was modified in the areas of active arm control, capture, and validation. Further work is required to interface the latest version of *ROCKET* with the thermal vacuum chamber hardware and software. A formal validation process must be performed to certify the new POHS modules and digital Servo Power Amplifier model.

## Reference 14

DCI TM#020499-1  
RMS Simulation CBM Verification Testing  
February 4, 1999

## Dynamic Concepts Technical Memorandum

#020499-1

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts - Dr. Patrick Tobbe  
Subject : RMS Simulation CBM Verification Testing

### Introduction

This memorandum describes the efforts made to date in validating *ROCKET* after the incorporation of the Position Orientation Hold Select (POHS) control system and the digital Servo Power Amplifier (SPA) models. These activities were necessary to meet the requirements of integrated CBM test 275.

### ROCKET Validation Results

The *ROCKET* simulation has been modified in three major areas of operation : active arm control, capture, validation. All three of these simulation modes now run under flight software control as documented in Functional Subsystem Software Requirements (FSSR) OI-26. Initialization and operating routines were substantially changed for all three of these simulation modes. This work was previously documented in DCI Technical Memorandum # 103198 -1.

Upon completion of the necessary software changes, *ROCKET* was first tested by running a subset of the earlier flight to simulation validation cases. The results were comparable to the original runs and flushed out software integration errors. These runs exercised the validation scheduler, the digital SPA's, and POHS controller modules. The arm was driven under the direct drive, single joint, and manual augmented modes of operation.

Next, mode transition logic was tested through inputs from the graphical user interface panel. The arm was initialized in manual augmented orbiter loaded mode and driven through the joysticks. The resulting motion was verified to be in the proper direction. The arm was allowed to transition into position hold and then switched to single joint mode. It was driven through several joints, one at a time, and brought to rest using the brakes and safing. Motion was again checked using the digital displays on the graphical user interface. At this point, the software was exercised by a group of Astronauts for the upcoming space station assembly missions. Based on their input, and that of the flight instructors, the fly-to fly-from modes were modified to incorporate reference frame changes from the standard orbiter frames. This software has been added and tested as an initialization feature. It will have to be modified in the future to interface to the personal computer which provides real time interaction with the RMS operator.

Compensation tests were also performed in the 6DOF facility using the latest two body software at a cycle of 2 milliseconds. These new curves must be integrated into *ROCKET* for the appropriate payloads. New curves will also be needed for the V20 chamber.

Formal Simulation to Simulation (S2S) tests were begun on *ROCKET* for four different payloads : 0K, 32K, 180K, 586K. The validation event scheduler was configured

to run the required eight cases for each of the four payloads. These cases consist of brakes on / thruster firings, direct drive, single joint, and manual augmented mode maneuvers. New thruster firing inputs files were created for the heavy payloads with appropriate software changes to utilize this data. The firing duration for the unloaded cases was also lengthened. The output routines were consolidated into a single output file. After initial runs were completed, discrepancies were discovered with the POR motion for the heavy payloads. This was traced to an error in a transformation matrix in the S2S validation procedures from JSC. The validation initialization algorithm was also generalized to accommodate different arm configurations. These changes improved the overall validation grade; however, there are still problems with the heavy payload, thruster firing cases.

In these cases, the brakes are slipping in the *ROCKET* simulation and not in the SPAR *ASAD* simulation. For the light payloads, the S2S comparisons show the brakes are holding. Upon further investigation, an error was discovered in the subroutine *LDASP* which transformed the joint freeplay stiffness matrices into inertial coordinates. This correction was made to *ROCKET* as well as *ACDS*. However, after rerunning these cases, the brakes were found to still be slipping. The *PAYLOAD* and *BASE* subroutines were reviewed and no problems were found. Parameter studies were run varying payload mass, inertia, and orientation. The simulation results were, as expected, sensitive to payload mass and inertia, but not orientation. The brake friction was also increased to hold the joints in place. However, these changes cannot be implemented since the values used are defined by the S2S validation process. The results can also be improved by softening the off axis freeplay stiffnesses. These values are defined in the validation document and can be logically modified to increase the validation grade. Further studies are required to determine if this is indeed feasible.

## **Conclusions / Recommendations**

Initial tests and validation cases have shown that the digital SPA models and POHS control system have successfully been integrated into *ROCKET*. However, these same tests have revealed a difference in the holding capability of the joint brakes between the two simulations. Since *ROCKET* appears to slip for lower joint torques, this presents a potentially serious problem for previous CBM analysis. Current analysis shows that the RMS brakes will slip during CBM berthing. If *ROCKET* is modified to match *ASAD*, the CBM latches will require more torque to complete, if possible, berthing operations. The RMS backdrive forces will also increase. For these reasons, it is imperative to understand the differences between *ASAD* and *ROCKET*, especially pertaining to CBM operations.

bd Systems®  
TCD20000103A  
30 April 2000

Contract No.  
NAS8-40604  
Final Technical Report

## Reference 15

DCI TM#052599-1  
RMS Simulation Support  
May 25, 1999

## Dynamic Concepts Technical Memorandum

#052599-1

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts - Dr. Patrick Tobbe, Mr. Ashley Hill  
Subject : RMS Simulation Support

### Introduction

This memorandum describes the technical support provided by Dynamic Concepts in support of the real time RMS simulation *ROCKET*. Analytical modeling and simulation software development activities were performed in the areas of enhanced man-in-the-loop (MIL) operation and formal simulation validation. No SSRMS math model development tasks were worked in this reporting period. Pending clarification of SSRMS model requirements, these tasks will be started in June, 1999.

### Enhanced MIL Operation

In preparation for ECP 275, three major modifications were made to *ROCKET* for MIL operation : addition of analytical contact force models, integration of an event scheduler / paper pilot, and calculation of RSAD and spec panel 169 display variables. New compensation curves based on 2 millisecond data using SGI hosted math models replaced earlier curves. The subroutine *COMP* was modified to account for changes in the payload mass properties library. Support was also provided to test the V20 option of the two body simulation, update the post processing mode of operation, and minimize the simulation cycle time.

### Contact Force Models

Three analytical contact force models have been added to the *ROCKET* software. These models include the guide-to-guide, thermal standoff, and Ready-To-Latch (RTL) mechanism contact force models. The contact force models are called from the dynamics loop routine *CONFM* that is in turn called by *RMSPLANT*. Flags are set by the user at initialization time in the *INTERACT* routine to determine whether to include the effects of the three models or not; values of 1 and 0 turn the contact force models on and off, respectively. *CON.GG\_CF\_FLAG* is the guide-to-guide contact force model flag, *CON.RTL\_CF\_FLAG* is the RTL contact force model flag, and *CON.PL\_CF\_FLAG* is the thermal standoff contact force model flag. If the RTL contact forces are to be included, the user is also given the choice of "flight" or "6-dof" RTL configurations. The difference in the two RTL's is that the "6-dof" RTL's protrude two inches higher than the "flight" RTL's.

The forces and torques output from these models are integrated into the total sensor force, FS1S1, and torque, TS1S1. Force and torque deadbands have also been added to the *ROCKET* software only mode logic. These deadbands have the same values as those employed by *ROCKET* during the HWIL simulation runs.

## Event Scheduler

*ROCKET* was also modified to include a new event scheduler for use as a "paper pilot" to command the regular pilot in the loop (PITL) commands. This software allows the user to define a set of events that may occur during any RMS maneuver. The events are read from an event scheduler input file located in the /paper\_pilot directory. The name of the event scheduler input file is input by the user in the *INTERACT* routine. An example of the event scheduler input file is given below. As shown in the table, six inputs are required to define an event: (1) the event number, (2) the type of event (continuous or roaming), (3) the variable to check the criteria for the event, (4) the value of the criteria variable needed for event occurrence, (5) the action to take upon event occurrence, and (6) an input needed dependent on the action. Of special note is the fact that the event scheduler input file is a formatted file; the format for each data field is given on line 2 of the file.

| Event Num | Event Type | Event Var | Event Var Val | Event Action | Action Input |
|-----------|------------|-----------|---------------|--------------|--------------|
| I2        | I2         | I3        | E13.6         | I3           | A15          |
| 1         | 1          | 1         | 0.0E0         | 6            | hc_lab.dat   |
| 2         | 1          | 1         | 0.0E0         | -13          |              |
| 3         | 1          | 1         | 160.D0        | 12           |              |

### Example Event Scheduler Input File

The event number is used for clarity and for output purposes only. An event type of "1" signifies a continuous event; these events occur only in the order they are input. In other words the event sequence for continuous events would be Event 1, Event 2, Event 3, etc. An event type of "2" signifies a roaming event; these events can happen any time once the event prior to it in the schedule occurs. For example, if Event 2 is a roaming event and all others are continuous events, a potential sequence could be Event 1, Event 3, Event 4, Event 2, Event 5.

The available criteria variables are given in Table 1. These variables represent most of the simulation variables that could be examined in order to perform a PITL action. The criteria variable number requires a sign ( $\pm$ ) for input in order for the event to properly occur. A positive criteria variable number signifies that the criteria variable must be greater than the input value before event occurrence, while a negative criteria variable number means that the criteria variable must fall below the input value for the event to occur.

The event actions consist of most of the actions available to the pilot; a list of these actions is given in Table 2. The user should be aware that some actions require a sign ( $\pm$ ) to properly function. Some event actions require further inputs that make up the sixth event definition input. For events that command a manually augmented mode, the action input is the name of the hand controller input file to be used. The hand controller input file is located in the /paper\_pilot directory and contains the counts for all translation and rotation inputs. For events that command either a single joint mode or direct drive mode, the action input is the number of the joint to be commanded.



**Table 1. Possible Event Criteria Variables**

| #  | Event Criteria Variable        |
|----|--------------------------------|
| 1  | Time                           |
| 2  | Time since last event          |
| 3  | RD2D1D1 X                      |
| 4  | RD2D1D1 Y                      |
| 5  | RD2D1D1 Z                      |
| 6  | D2PPD2 Euler angle Roll        |
| 7  | D2PPD2 Euler angle Pitch       |
| 8  | D2PPD2 Euler angle Yaw         |
| 9  | RTL Indicators (3 or more lit) |
| 10 | Latch #1 driver angle          |
| 11 | Latch #2 driver angle          |
| 12 | Latch #3 driver angle          |
| 13 | Latch #4 driver angle          |

**Table 2. Possible Event Actions**

| #  | Event Action                                       |
|----|--|
| 1  | Idle Mode  |
| 2  | Test Mode  |
| 3  | Single Joint Mode                                  |
| 4  | Direct Drive Mode                                  |
| 5  | Orbiter Unloaded                                   |
| 6  | Orbiter Loaded                                     |
| 7  | Payload Mode                                       |
| 8  | End Effector Mode                                  |
| 9  | All Brakes On                                      |
| 10 | Safing Mode  |
| 11 | Command Zero Change in Single / Direct Drive Angle |
| 12 | Terminate Run                                      |
| 13 | Toggle Coarse (+) / Vernier (-) Scaling            |
| 14 | Command +/- Single or Direct Drive Angle           |
| 15 | Zero Hand Controller Inputs                        |
| 16 | Turn Latches Off/On                                |
| 17 | Toggle Rate Hold Request Off/On                    |

In order to illustrate the use of both the new analytical contact force models and the paper pilot mode of operation in *ROCKET*, the results of a set of simulation runs incorporating the new features will be presented. The event scheduler input file given above was used in both HWIL and software only modes of *ROCKET* to drive the active ring

into the passive ring until motion stalled. As shown in the event scheduler input file, Event 1 commands Orbiter Loaded Mode at time zero and hand controller inputs are read from *hc\_lab.dat*. Event 2 commands Vernier scaling rate mode at time zero, and Event 3 terminates the simulation at a time of 160 seconds. The HWIL run employed the actual guides and utilized the analytical thermal standoff and RTL contact models, while the software only runs used the analytical guide-to-guide, thermal standoff, and RTL contact models. The software only runs had different guide surface friction coefficients to determine the best value to match the actual guides used in the HWIL run. The relative position and orientation results for all the *ROCKET* runs are given in the attached plots. As shown in the plots, the software only run with a friction coefficient of 0.01 matches the HWIL run pretty well. These results reveal not only that the paper pilot runs can be used to rerun the same set of pilot actions repeatedly for different simulation parameters, but also that the contact force models adequately simulate the contact forces and torques.

### RSAD Outputs

Output variables have been added to *ROCKET* to support the RSAD data packet and PDRS Status (DISP 169) displays during PITL operation. These output variables will be placed in a structure within *ROCKET* in order to meet the RSAD and display communication requirements. The *ROCKET* routines *KDG* and *DDP* were modified per the RMS FSSR to calculate some of the of output variables not already output in the simulation. The variables output to RSAD and the PDRS Status display are given in Table 3 below.

**Table 3. RSAD and Display Outputs**

| RSAD Packet Outputs         | Description   | Subroutine      |
|-----------------------------|---|-----------------|
| ROT_EE_ORAS                 | Rotation from ORAS Frame to End Effector Frame Quaternion | <i>KDG</i>      |
| TRANS_EE_ORAS               | End Effector Position in ORAS Frame (in)                  | <i>KDG</i>      |
| PDRS Status Display Outputs | Description   | Subroutine      |
| ACT_POR_ROT_RATE_SEL        | POR Actual Rotation Rate (deg/sec)                        | <i>KDG</i>      |
| ACT_POR_TRANS_RATE_SEL      | POR Actual Translation Rate (ft/sec)                      | <i>KDG</i>      |
| JOINT_ANGLES_DISPLAY        | Joint Angles for Display (deg)                            | <i>KDG</i>      |
| POHS_REF_ATT_STR            | POHS Reference POR Attitude (deg)                         | <i>KDG</i>      |
| POHS_REF_POS_DISP           | POHS Reference POR Position (in)                          | <i>KDG</i>      |
| POR_ATT_STR                 | POR Attitude Display (deg)                                | <i>KDG</i>      |
| POR_POS_DISP                | POR Position Display (in)                                 | <i>KDG</i>      |
| POR_ROT_RATE_CMD_DISP       | POR Rotation Rate Commands (deg/sec)                      | <i>DDP</i>      |
| POR_TRANS_RATE_CMD_DISP     | POR Translation Rate Commands (ft/sec)                    | <i>DDP</i>      |
| REL_POR_REF_SEL             | Relative POR Reference Display Flag                       | <i>INTERACT</i> |

## **ROCKET Validation Support**

Initial validation efforts centered on solving the brake slip discrepancy reported earlier in DCI Technical Memorandum # 020499-1. The *ROCKET* simulation was showing brake slip for the thruster firing heavy payload validation cases while the *ASAD* results did not. Eventually this was traced to a formatted read error in the thruster firing loads files. *ROCKET* and *ASAD* correlated well for thruster firing cases upon correction of this mistake.

The remaining validation runs (direct drive, single joint, MIL) were made for the four payloads. While the results were not bad, there is room for significant improvement in the POR motion. The POR equations to include flexible body deformations have been reviewed and checked against rigid body motion. However, POR results are strongly influenced by structural model differences between *ASAD* and *ROCKET*. An effort is currently underway to investigate model differences in the joint freeplay degrees of freedom and spring constants used by the two simulations. Data is also being reviewed from the *TRICK* and *SOMBAT* simulations. In particular, the effects of different arm linkage dimensions used by the flight software and arm dynamics models on simulation results are being studied. Part of the problem lies in the lack of output variable definitions from *ASAD*. New sliding equilibrium point (SEP) joint friction models have been coded to evaluate the effects of joint friction on gear box torques and joint motion.

## **Conclusions / Recommendations**

Several modifications were made to *ROCKET* to enhance MIL operations and improve the formal validation results. MIL operation now includes optional analytical contact force models, an event controller or paper pilot, and several new display variables. Future MIL activity will focus on the development of a separate MIL process and integration with the new spec panels.

Work will continue toward improving the validation simulation results. Efforts will be focused on joint friction and freeplay stiffness parameter studies. Documentation of the *TRICK* and *SOMBAT* simulations will be reviewed for any input data or modeling differences with *ROCKET*.

bd Systems®  
TCD20000103A  
30 April 2000

Contract No.  
NAS8-40604  
Final Technical Report

## Reference 16

DCI TM#122799-1  
RMS Simulation Support  
December 27, 1999

## Dynamic Concepts Technical Memorandum

#122799-1

To: bd Systems - Mr. Ronald Francis  
From : Dynamic Concepts -Dr. Patrick Tobbe, Dr. Thomas Howsman,  
Mr. Ashley Hill, Mr. Mike Craft  
Subject : RMS Simulation Support

### Introduction

This memorandum describes the technical support provided by Dynamic Concepts in support of the real time simulations *ROCKET* and *2BODY*. Activities were performed to enhance the *ROCKET* validation results, direct drive operation, and hardware-in-the-loop (HWIL) stability. *2BODY* was upgraded to include CBM contact force models and expand compensation testing capability.

### ***ROCKET* Validation Activities**

After completion of the formal validation process, *ROCKET* graded out at 87%. Although these results are consistent with the performance of other simulations, the Point of Resolution (POR) responses are erratic. Upon further investigation, it was discovered that the POR motion was highly dependant on the relinearization frequency or update rate. It should be noted that the total POR motion, including substructure and joint deformations, is computed from linearized matrices which are updated periodically. The effects of the approximations used to compute these matrices are more pronounced when large amplitude arm motion occurs between the updates. Since the POR motion is the sum of the state at the relinearization time and perturbations away from this configuration, the error will accumulate with time. These errors can be minimized by decreasing the time between updates. POR results were dramatically improved when the relinearization time interval was decreased from 4 seconds to .1 seconds. However, *ROCKET* cannot run in real time for this update rate. Another solution to this problem is to compute the actual POR motion directly without the use of linearized matrices. Ideally, this could be done every time step. However, computation constraints may make this prohibitive for real time operation. The actual POR motion could be computed at the relinearization frequency, with the results used to drive the accumulated POR errors to zero. This should be investigated in future studies.

A review of SRMS math models and validation criteria is currently being performed by the SRMS Math Model Working Group (MMWG). This effort is concentrating on understanding the differences between the SPAR *ASAD* model and the *TRICK* simulation. Upon resolution of these differences, the MMWG will select common algorithms for the various SRMS simulation modules to be used by the community. MSFC and *ROCKET* should be involved in this process. *ROCKET* can be used to help identify the differences between *TRICK* and *ASAD*. This may

give the RMS trainers and astronauts more confidence in *ROCKET*. The next MMWG meeting is scheduled for sometime in January, 2000.

### ***ROCKET* Direct Drive Operation Upgrade**

The direct drive mode of operation was modified based on inputs from the JSC RMS operator trainers. According to the JSC trainers, the brakes must be applied and the RMS mode switch set to "Direct" before the direct drive mode is active. In direct drive mode the brakes are applied to all joints except the one joint being driven with the single joint/direct drive toggle switch. When the toggle switch is released, the brakes are then applied to the driven joint as well. The routines affected by this change include *CTLINIT*, *MOTOR*, *RMS*, and *S\_DYN.INC*.

### ***ROCKET* HWIL Stability Tasks**

Two areas were examined in an effort to stabilize HWIL simulations with *ROCKET* during CBM bolting operations and deberthing studies. A numerical integration study was performed using the Analytical Contact Dynamics Simulation (*ACDS*) tool to compare the Rectangular and Runge-Kutta methods. The Table Contact Dynamics Simulation (*TCDS*) tool was used to evaluate artificial payload damping and contact force filtering techniques.

#### ***ACDS* Numerical Integration Study**

*ACDS* was used to perform an initial study comparing the Rectangular and Runge-Kutta integration methods. To provide a basis for comparison, a modified version of *ACDS* was created which implemented a fourth order Runge-Kutta (RK4) integration algorithm. In this study, the RMS joints were disconnected from the servo motors and were free to rotate. The contact force model was replaced by a constant external load applied at the S1 sensor location. All of the boom substructure modes were retained. Using configuration 17, a series of runs were made with both the standard *ACDS* and the *ACDS*/RK4 simulation using progressively smaller step sizes until the RMS response effectively converged. The simulation run time was 30 seconds and relinearization was not active. Simulation parameters which were studied to determine convergence were the position vector from the base vehicle docking port to the module docking port (*RD2D1D1*), the relative velocity vector between the base vehicle docking port and the module docking port (*RD2D1D1D*), and the CBM relative Euler angles. Once an acceptable benchmark was created, the simulation integration step size was increased until either the simulation results were unacceptable or the simulation became unstable.

A total of eight simulation runs were performed; four using the Rectangular integration method and four using the Runge-Kutta integration method. Acceptable convergence of simulation results was found using an integration step size of

.00002 sec. For the purposes of this study, the baseline results are considered to be those generated by ACDS using Rectangular integration at an integration step size of .00002 sec. The ACDS input file used to generate the baseline data is presented in Figure 1 for reference. Diagrams showing a comparison of the baseline RMS simulation parameters of interest to those generated by the ACDS simulation using the RK4 algorithm at an integration step size of .00002 sec are presented in Figures 2 through 10.

```

ct4_bench_1      ! Root Name of Output Files
1                ! MODEOP: Capture Mode Selected
2                ! Capture Mode(1=Idle,2=Test,3=POHS,4=Joint POS_HOLD,5=Dynamic)
1                ! flag for use of OUTPUTC results(0-off,1-on)
0.00002d0        ! dt rms model
0.00002d0        ! dt hardware (latches) (sec)
30.d0            ! simulation run time (sec)
400              ! print rate (mprn*dt -> print time)
17              ! payload & base vehicle number
false false      ! payload flex, base flex flags
1.0 1.0          ! payload mass and inertia factors
0               ! end effector flag (0-off,1-on)
0               ! alignment pin contact model (0-off,1-on)
0               ! Duckhead bumper flag (0/1 off/on)
0.d0            ! 6dof RD1D2D2(1) offset - seal bead height (in)
0 0 0 0 0       ! contact models(1/0) r-r, ra-gb, rb-ga, g-g, plunger
0               ! flag (1/0) for latch enactment
0               ! flag (1/0) for RTL contact force model
30.             ! Lowpass Frequency Limit (rad/sec)
0               ! Force/Moment Deadband flag (0-off,1-on)
17, 2, 1, 0,    ! dal unit, cycle #s (0,0 -> no output)
5.5, 0.8, -0.8  ! vector RD1D2D2 (units= inches)
-0.2, 0.2, -0.2 ! EA(1), EA(2), EA(3) (units= degrees)
0.0, 0.0, 0.0   ! vector RD1D2D2D (units= ft/sec)
0.0, 0.0, 0.0   ! angular velocity vector (units= degrees/sec)
1               ! Payload Fly Mode (1-Nominal,2-Fly)
1               ! sequential output (1=yes,2=no)
2               ! SPA Control Selection (1-Analog,2-Digital)
2               ! POHS Control Selection (1-Disabled,2-Enabled)
0               ! BERTH Graphics output flag (0/1 no/yes)

```

Figure 1: Rectangular vs. RK4 Integration Method Study  
Baseline Simulation Input File

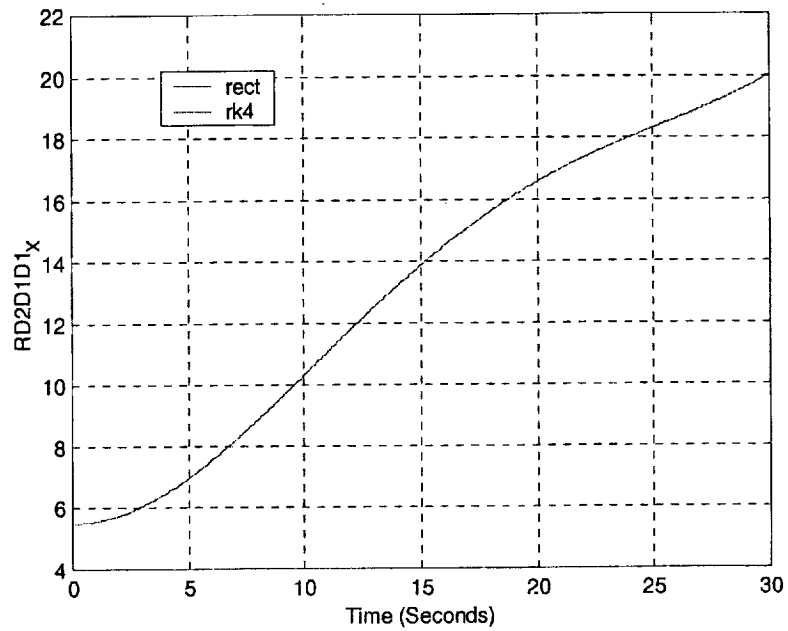


Figure 2: Rectangular vs. RK4 Integration Method Baseline Comparison  
 $RD2D1D1_x$  (inches) Time History,  $dt=.00002$  seconds

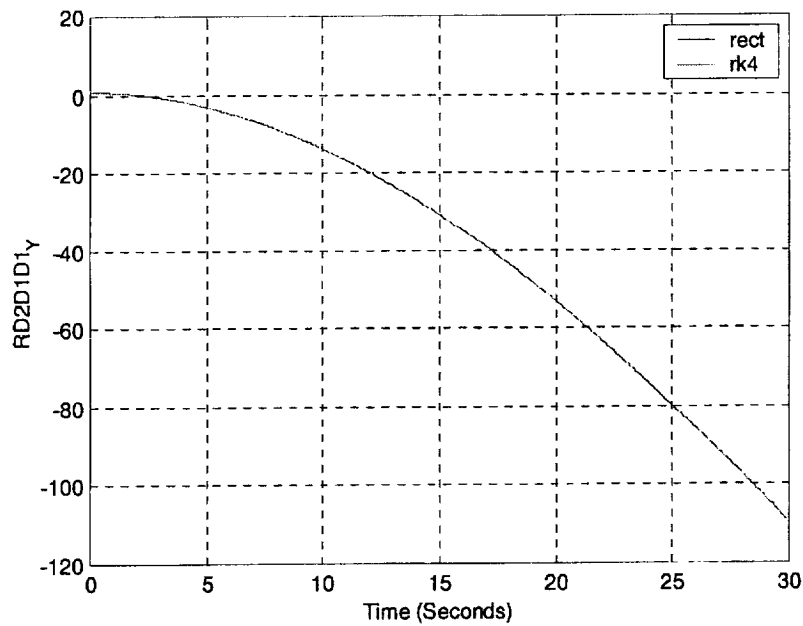


Figure 3: Rectangular vs. RK4 Integration Method Baseline Comparison  
 $RD2D1D1_y$  (inches) Time History,  $dt=.00002$  seconds



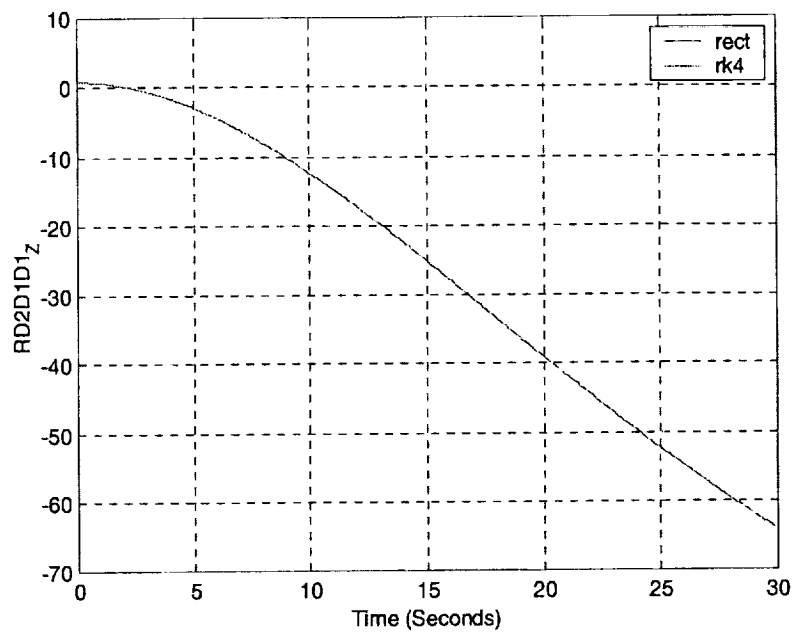


Figure 4: Rectangular vs. RK4 Integration Method Baseline Comparison  
 $RD2D1D1_z$  (inches) Time History,  $dt=.00002$  seconds

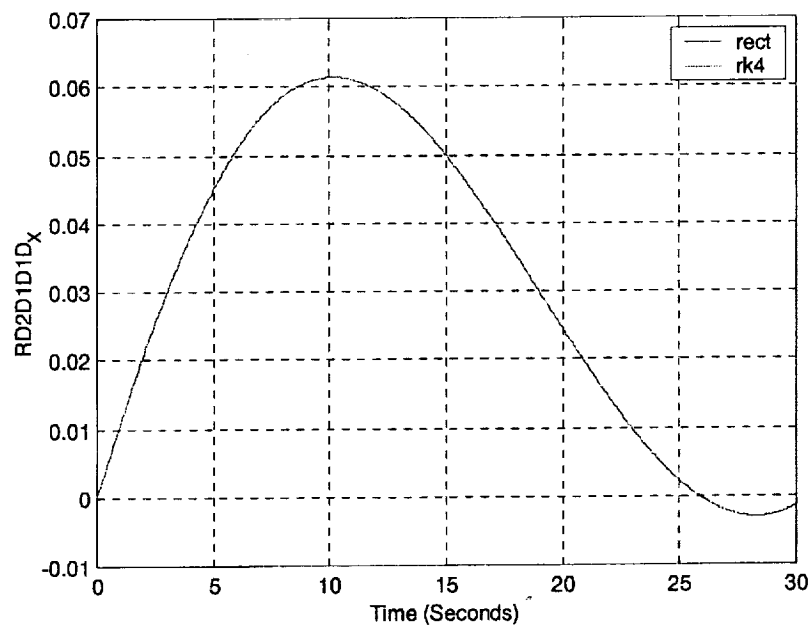


Figure 5: Rectangular vs. RK4 Integration Method Baseline Comparison  
 $RD2D1D1D_x$  (inches/sec) Time History,  $dt=.00002$  seconds

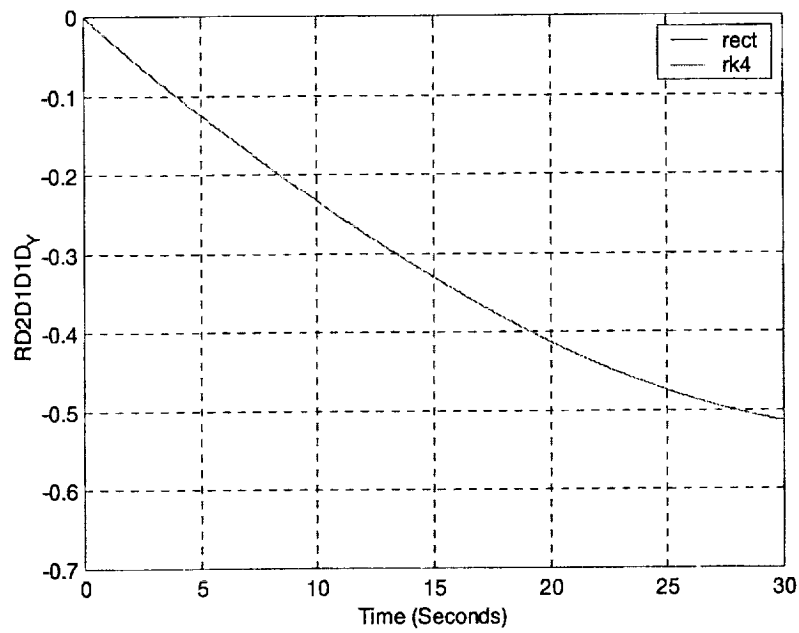


Figure 6: Rectangular vs. RK4 Integration Method Baseline Comparison  
 $RD2D1D1D_y$  (inches/sec) Time History,  $dt=.00002$  seconds

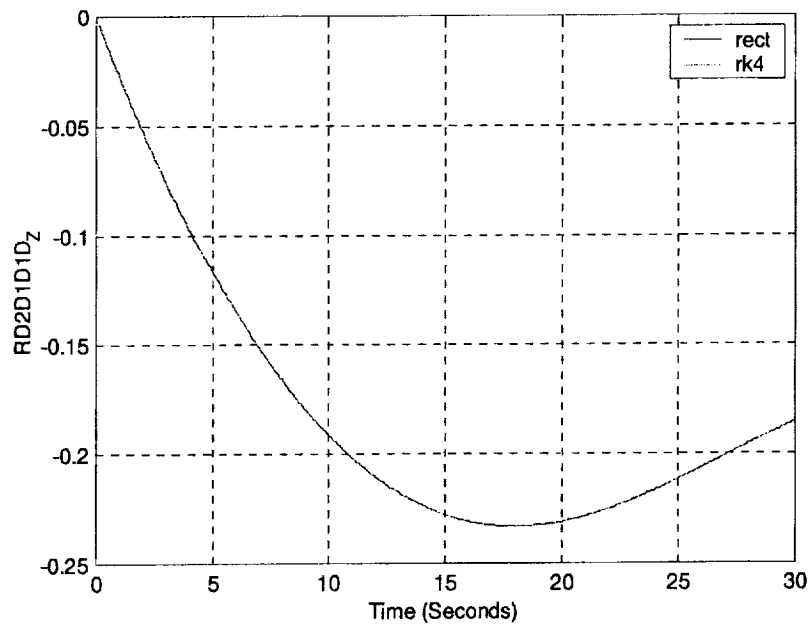


Figure 7: Rectangular vs. RK4 Integration Method Baseline Comparison  
 $RD2D1D1D_z$  (inches/sec) Time History,  $dt=.00002$  seconds

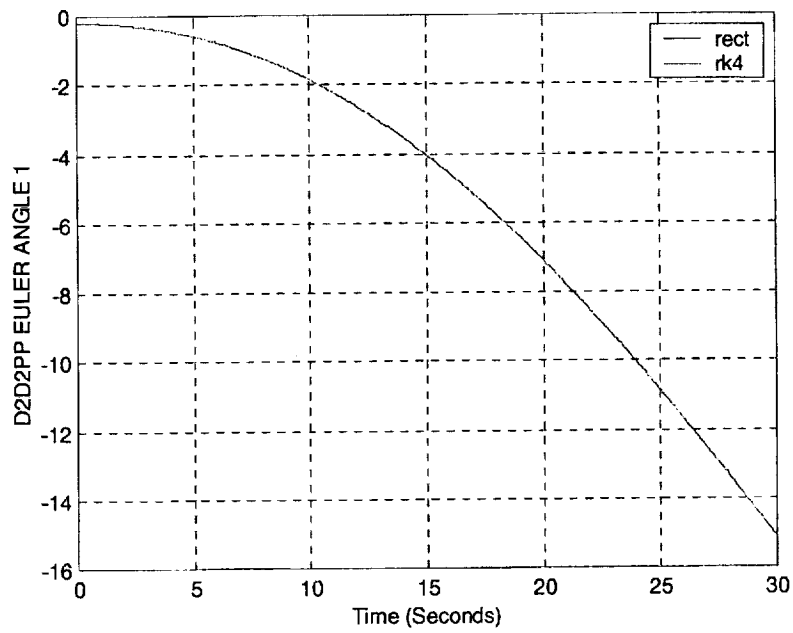


Figure 8: Rectangular vs. RK4 Integration Method Baseline Comparison  
Euler Angle 1 (degrees) Time History,  $dt=.00002$  seconds

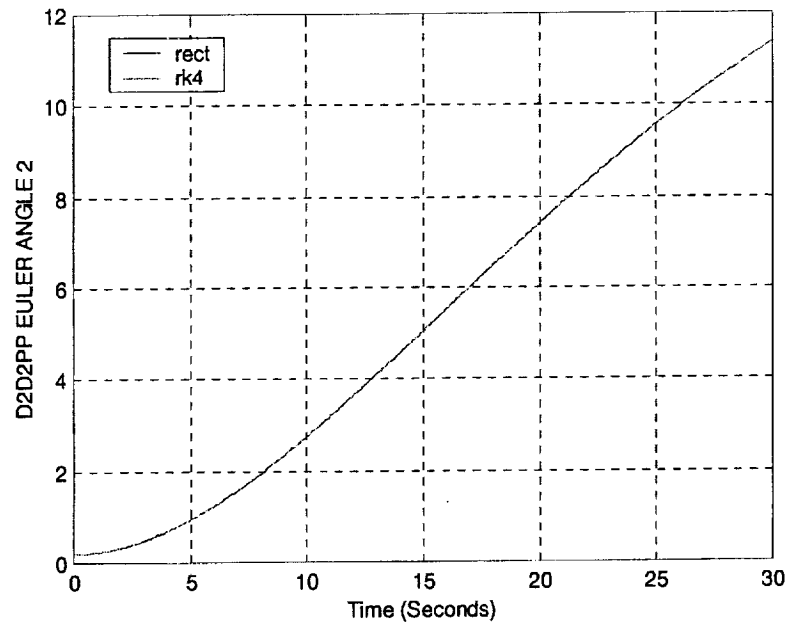


Figure 9: Rectangular vs. RK4 Integration Method Baseline Comparison  
Euler Angle 2 (degrees) Time History,  $dt=.00002$  seconds

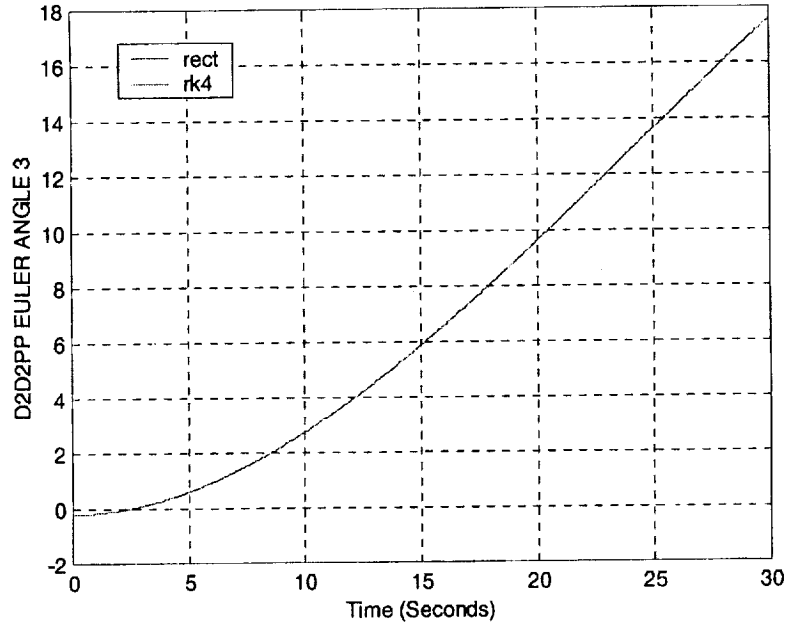


Figure 10: Rectangular vs. RK4 Integration Method Baseline Comparison  
Euler Angle 3 (degrees) Time History, dt=.00002 seconds

The integral of the percent error of each of the simulations' parameters was calculated to provide a single, concise term that gives insight into the degree to which the simulation results match the baseline. The error integral term is expressed as

$$Error\ Integral = \int_0^T \left[ \frac{(\alpha_s - \alpha_b)}{\alpha_b} * 100 \right] dt$$

Where  $\alpha_b$  = baseline simulation parameter  
 $\alpha_s$  = comparison simulation parameter

A comparison of the error integrals calculated for each simulation run is presented in Table 1, as well as the approximate clock time required to execute each simulation. All simulation runs were calculated on a 450MHz PII computer. However, the reader should regard the execution times presented in Table 1 as a general approximation rather than an exact time since other applications were running on the computer while the simulations were executing. The simulation integration step size of Rectangular integration Run 4 is .01 seconds rather than .02 seconds because the simulation became numerically unstable when the step size was above .01 seconds.

Table 1: Rectangular vs. RK4 Integration Method Results Comparison

|                           | RK4 Integration |         |         |         | Rectangular Integration |         |         |
|---------------------------|-----------------|---------|---------|---------|-------------------------|---------|---------|
|                           | 1               | 2       | 3       | 4       | 2                       | 3       | 4       |
| Integration Step (sec)    | 0.00002         | 0.0002  | 0.002   | 0.02    | 0.0002                  | 0.002   | 0.01    |
| Simulation Run Time (sec) | 1509.27         | 148.30  | 14.96   | 1.72    | 42.70                   | 4.44    | 1.08    |
| Error Integral            |                 |         |         |         |                         |         |         |
| RD2D1D1 X                 | 0.00346         | 0.00346 | 0.00346 | 0.00346 | 0.03120                 | 0.34313 | 1.71878 |
| RD2D1D1 Y                 | 0.00710         | 0.00710 | 0.00710 | 0.00710 | 0.06400                 | 0.70389 | 3.54349 |
| RD2D1D1 Z                 | 0.00673         | 0.00673 | 0.00673 | 0.00673 | 0.06047                 | 0.66513 | 3.34951 |
| RD2D1D1D X                | 0.00350         | 0.00350 | 0.00350 | 0.00353 | 0.03160                 | 0.35100 | 2.14770 |
| RD2D1D1D Y                | 0.00153         | 0.00153 | 0.00153 | 0.00154 | 0.01368                 | 0.15477 | 1.16731 |
| RD2D1D1D Z                | 0.00150         | 0.00150 | 0.00150 | 0.00151 | 0.01351                 | 0.15033 | 1.11602 |
| EULER ANGLE 1             | 0.00375         | 0.00375 | 0.00375 | 0.00375 | 0.03374                 | 0.37117 | 1.87058 |
| EULER ANGLE 2             | 0.00245         | 0.00245 | 0.00245 | 0.00245 | 0.02204                 | 0.24244 | 1.22238 |
| EULER ANGLE 3             | 0.00423         | 0.00423 | 0.00423 | 0.00422 | 0.03808                 | 0.41890 | 2.11105 |

Results of the study show that, under the special conditions required to perform the comparison, the Runge-Kutta integration method performed well. The error integrals associated with the simulation parameters generated by the RK4 method show no significant increase due to integration step size for the range of integration steps covered by this study. When Rectangular integration was used, the error integrals increased significantly as the integration step size increased. In general, the quality of the simulation results was higher using the RK4 integration method. Error integrals of the simulation with the highest integration step size using the RK4 integration method (RK4 Run 4) are approximately an order of magnitude less than those of the simulation using the Rectangular integration method with the lowest integration step size (Rectangular Run 2). However, it should be noted that when the time histories of parameters associated with this study's "worst" simulation results (Rectangular Run 4) are directly compared to baseline results, the curves appear nearly identical.

While the results of this study show that the Runge-Kutta integration algorithm works well with *ACDS*, it is necessary to report that the integration algorithm developed for this study yields acceptable results only for the conditions under which this study was conducted. For instance, when the effects of gear train torque and motor back drive torque were included in the simulation, results generated by the *ACDS*/RK4 simulation did not match those generated by the *ACDS* simulation using Rectangular integration. It is apparent that there are fundamental issues concerning the interaction of the Runge-Kutta integration algorithm, the algorithm that calculates the time derivative of the simulation state vectors (*RMSPLANT*), and the algorithm that calculates the perturbation variables (*INTFAC*) that should be addressed. These issues were not completely examined due to the time constraints associated with this study. However, if it is desired to fully implement the RK4 integration algorithm into *ACDS*, it will be necessary to

consider modification to the design and implementation of the integration, *RMSPLANT*, and *INTFAC* algorithms.

### ***TCDS* Payload Damping Evaluation**

In order to stabilize the V20 chamber when driven by *ROCKET* during CBM bolt operations, a combination payload damping and force filtering scheme was implemented. This approach applies an analytical damping force and moment to the system based on the relative motion between the active and passive rings. The damping force and moment vectors are computed as

$$\mathbf{F}_{DAMP} = 2 \zeta \omega m \dot{\mathbf{R}}_{D2D1}$$

$$\mathbf{T}_{DAMP} = 2 \zeta \omega I \dot{\boldsymbol{\Theta}}_{D2D1}$$

where  $m$  is the payload mass,  $I$  is the diagonal element of the payload inertia matrix in the payload structure frame,  $\zeta$  is the damping ratio,  $\omega$  is the damping frequency,  $\dot{\mathbf{R}}_{D2D1}$  is the relative translational velocity, and  $\dot{\boldsymbol{\Theta}}_{D2D1}$  is the relative angular velocity. The damping frequency is a constant 0.10 rad/sec for separation distances greater than 7 inches. At this point, it is linearly increased to a value of 50 rad/sec at a separation distance of 2 inches. The damping ratio is a constant 0.30. These values were the result of analytical parameter studies in *TCDS* and HWIL capture runs with *ROCKET*. The damping frequency is not currently allowed to decrease if the x component of the separation vector increases. These values, along with force filtering, stabilize HWIL runs with unity mass and inertia factors with a frequency limit of 30 rad/sec for all payloads tested. However, resulting CBM relative lateral motion and wobble angles were significantly smaller in these HWIL runs than those predicted by *ACDS* with no damping or filtering. The force and moment damping vectors are computed in the D1 frame and transformed to the S1 frame. These vectors are then added to the filtered contact loads *FS1S1* and *TS1S1*, with the sum being applied to the RMS.

The force filtering routine is a second order low pass digital filter with a cut off frequency of 2 Hertz. This filter is applied to the analytical and measured contact loads, not the payload damping terms. Details of the filter software are presented in the following paragraphs.

The files *s\_con.inc*, *s\_con.h*, *interact.f*, *start.f*, and *rmsplant.f* were modified to incorporate the payload damping and force filtering algorithm into *ROCKET*. *interact* prompts the user to initiate damping through the variable *con.pyld\_damp\_flag*. The user then enters the value of the damping frequency for a separation distance of 2 inches, *con.w\_damp\_2in*. The force filtering algorithm

is enabled via *interact* through *con.f\_fil\_on*. The filter routines were added to the end of the files *start.f* and *rmsplant.f*.

A second-order digital IIR filter was designed to be an easily reconfigurable low-pass filter for the system. The recursive filter is modeled after the Butterworth family of analog filters. In general, the transfer function of a recursive filter is given as

$$H(z) = \frac{\sum_{k=0}^M a_k z^{-k}}{1 + \sum_{k=1}^L b_k z^{-k}}$$

which can be specialized for the 2<sup>nd</sup> order Butterworth filter to

$$y_n = a_0 x_n + a_1 x_{n-1} + a_2 x_{n-2} - b_1 y_{n-1} - b_2 y_{n-2}$$

where the  $y$  values are the filter outputs, and the  $x$ 's are filter inputs. The filter coefficients ( $a$ 's and  $b$ 's) are determined solely by the sample rate and the filter break frequency. A FORTRAN subroutine that can be used to initialize the filter is listed below.

---

```

      subroutine butter_2nd_compute(f_c, f_s, a, b)
c
c ... This subroutine computes the IIR filter associated with the
c ... 2nd Order Butterworth filter (low-pass).
c
c ... INPUTS
c ... f_c = cut off frequency [Hz]
c ... f_s = sampling frequency [Hz]
c
c ... OUTPUTS
c ... a = array of recursive coefficients
c ... b = array of recursive coefficients
c ... Usage:  y_n = a1*x_n + a2*x_n-1 + a3*x_n-2 - b1*y_n-1 - b2*y_n-2
c
      real*8 f_c , f_s
      real*8 a(3), b(2)

      real*8 Atemp, Btemp, Ctemp, Wtemp, pi

      pi = 4.d0 * tan( 1.d0 )

      Wtemp = tan( (f_c * 2.d0 * pi)/ (2.d0 * f_s) )

      Atemp = 1.d0 + 1.414d0*Wtemp + Wtemp*Wtemp
      Btemp = -2.d0 + 2.d0*Wtemp*Wtemp
      Ctemp = 1.d0 - 1.414d0*Wtemp + Wtemp*Wtemp

      a(1) = Wtemp*Wtemp/Atemp
      a(2) = 2.d0 * a(1)
      a(3) = a(1)

      b(1) = Btemp / Atemp
      b(2) = Ctemp / Atemp

      return
      end

```

---



Implementation of the recursive filter into the simulation is straightforward, requiring only successive calls to a subroutine that implements the recursion equation. An example of such a routine is given below.

---

```

subroutine butter_2nd_implement( a, b, x, y, y_n)
c
c ... This subroutine implements the IIR filter associated with the
c ... 2nd Order Butterworth filter (low-pass).

c ... INPUTS
c ... a = array of recursive coefficients
c ... b = array of recursive coefficients
c ... x = array of filter input values
c ... y = array of past filter output values

c ... OUTPUTS
c ... y_n = latest value of filter output (scalar)

c ... Usage:  y_n = a0*x_n + a1*x_n-1 + a2*x_n-2 - b1*y_n-1 - b2*y_n-2

c ... Note: x(1) = current input (x_n)
c          x(2) = filter input from previous step (x_n-1)
c          x(3) = filter input from 2 steps back (x_n-2)

c          y(1) = filter output from previous step (y_n-1)
c          y(2) = filter output from 2 steps back (y_n-2)

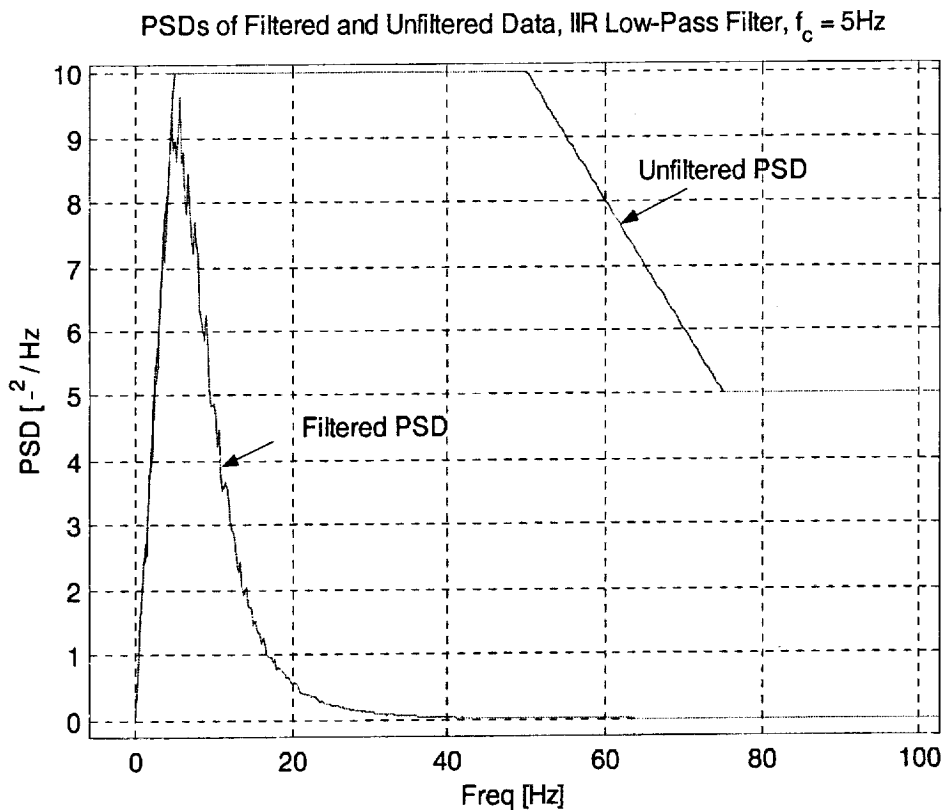
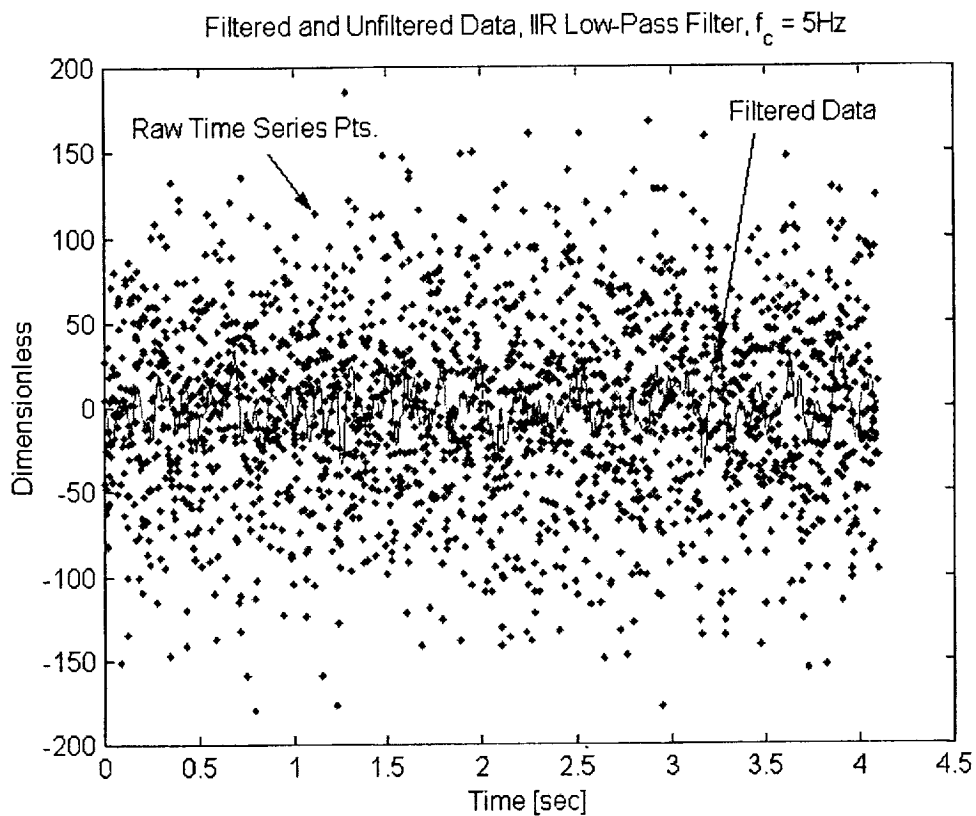
real*8 a(3), b(2)
real*8 x(3), y(2)

real*8 y_n

```

---

An example of the effectiveness of the digital filter is given in the following two figures. The first figure illustrates the discrete time histories of both the filter input and output. A cutoff frequency for the low-pass filter was set at 5Hz, and the data was sampled at 500Hz ( $\Delta t = 0.002$  sec). As can be seen, the higher frequency content associated with the filter input is greatly diminished in the output. The second figure illustrates this same information, albeit in the frequency domain. Clearly the output spectra is significantly reduced when compared to the input spectra for frequencies beyond that of the cutoff (5Hz).



## **2BODY Compensation Testing Modifications**

The compensation options of *2BODY* were modified to allow testing in all axes, one at a time. *start.f* was modified to query the user which degree of freedom, one through six, to initiate in table N or sensor S1 space. *fmtran.f* was changed to set to zero all measured loads except that of the table axis selected. *comp.f* was updated to apply the compensation in the user selected axis. For translational testing, the desired unit vector was transformed to D1 coordinates to select the proper velocity component for use in the compensation equations. *comp.f* was also modified for *ROCKET* to select the appropriate curves using values of the simulated mass and inertias instead of payload configuration numbers. The inertia matrices were transformed into table space N coordinates to match the curves with the proper axis. This reduced the number of rotation compensation functions to one and streamlined the routine.

Compensation tests were performed using the CBM hardware for all payload configurations. New curves were generated and coded into *comp.f*. CBM HWIL check runs were used to verify or modify these curves. Currently, the LAB, MPLM, and Z1 configurations all use the same curves. Only the PMA configurations use a different level of compensation.

## **2BODY Contact Force Models**

Three analytical contact force models have been added to the *2BODY* software. These models include the guide-to-guide, thermal standoff, and Ready-To-Latch (RTL) mechanism contact force models. The contact force models are called from the dynamics loop routine *CONFM* that is in turn called by *FMTRAN*. Flags are set by the user at initialization time in the *START* routine to determine whether to include the effects of the three models or not; values of 1 and 0 turn the contact force models on and off, respectively. *TB.GG\_CF\_FLAG* is the guide-to-guide contact force model flag, *TB.RTL\_CF\_FLAG* is the RTL contact force model flag, and *TB.PL\_CF\_FLAG* is the thermal standoff contact force model flag.

The forces and torques output from these models are integrated into the total sensor force, *F1CS1*, and torque, *T1CS1*. Force and torque deadbands have also been added to the *ROCKET* software only mode logic. These deadbands have the same values as those employed by *ROCKET* during the HWIL simulation runs.

## **Conclusions / Recommendations**

A variety of tasks have been performed in support of *ROCKET* and *2BODY*. Currently, work should continue in reducing the approximations used in the POR equations. Further analytical studies are required to determine the feasibility of the use of Runge-Kutta integration schemes in real time applications. Current SSRMS modeling efforts are focused on the procurement and integration of *TRICK* into the 6DOF facility.

| REPORT DOCUMENTATION PAGE  |                               |   | Form Approved<br>OMB No. 0704-0188 |                            |
|--|-------------------------------|---|------------------------------------|----------------------------|
| Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503. |                               |   |                                    |                            |
| 1. AGENCY USE ONLY (Leave blank)   | 2. REPORT DATE<br>30 April 00 | 3. REPORT TYPE AND DATES COVERED<br>FINAL / Nov 95 - April 00 |                                    |                            |
| 4. TITLE AND SUBTITLE<br>Final Report for Mathematical Model<br>Development and Simulation Support   |                               | 5. FUNDING NUMBERS<br>Contract:<br>NAS8-40604                 |                                    |                            |
| 6. AUTHOR(S)<br>Ronald C. Francis<br>Patrick A. Tobbe, Dr.   |                               | 8. PERFORMING ORGANIZATION<br>REPORT NUMBER                   |                                    |                            |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Control Dynamics, A Division of bd Systems, Inc.<br>600 Boulevard South, Suite 304<br>Huntsville, AL 35802   |                               | 10. SPONSORING/MONITORING<br>AGENCY REPORT NUMBER             |                                    |                            |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>National Aeronautics and Space Administration<br>George C. Marshall Space Flight Center<br>Marshall Space Flight Center, AL 35812   |                               | 11. SUPPLEMENTARY NOTES                                       |                                    |                            |
| 12a. DISTRIBUTION AVAILABILITY STATEMENT   |                               | 12b. DISTRIBUTION CODE  |                                    |                            |
| 13. ABSTRACT (Maximum 200 words)<br><br>This report summarizes the work performed in support of the<br>Contact Dynamics 6DOF Facility and the Flight Robotics Lab at NASA/<br>MSFC in the areas of Mathematical Model Development and<br>Simulation Support.   |                               |   |                                    |                            |
| 14. SUBJECT TERMS<br>Contact Dynamics Simulation Lab, Flight Robotics Lab,<br>6DOF, SRMS   |                               |   | 15. NUMBER OF PAGES<br>~ 300       |                            |
| 17. SECURITY CLASSIFICATION<br>OF REPORT<br>unclass  |                               |   | 16. PRICE CODE                     |                            |
| 18. SECURITY CLASSIFICATION<br>OF THIS PAGE<br>unclass   |                               | 19. SECURITY CLASSIFICATION<br>OF ABSTRACT<br>unclass         |                                    | 20. LIMITATION OF ABSTRACT |